Exhibit 15

Exhibit 15 Page 204

EIA
STANDARD

Digital Television (DTV) Closed Captioning

EIA-708-B
(Revision of EIA-708-A)

DECEMBER 1999

ELECTRONIC INDUSTRIES ALLIANCE

ENGINEERING DEPARTMENT

CEA
Consumer Electronics Association
A sector of EIA

EIA
Electronic Industries Alliance

Exhibit 15 Page 205

SONY0005722

NOTICE

EIA Engineering Standards and Publications are designed to serve the public interest through eliminating misunderstandings between manufacturers and purchasers, facilitating interchangeability and improvement of products, and assisting the purchaser in selecting and obtaining with minimum delay the proper product for his particular need. Existence of such Standards and Publications shall not in any respect preclude any member or nonmember of EIA from manufacturing or selling products not conforming to such Standards and Publications, nor shall the existence of such Standards and Publications preclude their voluntary use by those other than EIA members, whether the standard is to be used either domestically or internationally.

Standards and Publications are adopted by EIA in accordance with the American National Standards Institute (ANSI) patent policy. By such action, EIA does not assume any liability to any patent owner, nor does it assume any obligation whatever to parties adopting the Standard or Publication.

This EIA Standard is considered to have International Standardization implication, but the International Electrotechnical Commission activity has not progressed to the point where a valid comparison between the EIA Standard and the IEC document can be made.

This Standard does not purport to address all safety problems associated with its use or all applicable regulatory requirements. It is the responsibility of the user of this Standard to establish appropriate safety and health practices and to determine the applicability of regulatory limitations before its use.

(From Project Number 4749, formulated under the cognizance of the R-4.3 Television Data Systems Subcommittee.)

Published by

©ELECTRONIC INDUSTRIES ALLIANCE 1999
Engineering Department
2500 Wilson Boulevard
Arlington, VA 22201

PRICE: Please refer to the current
Catalog of EIA Electronic Industries Alliance Standards and Engineering Publications
or call Global Engineering Documents, USA and Canada (1-800-854-7179)
International (303-397-7956)

Exhibit 15 Page 206

PLEASE!

DON"T VIOLATE
THE
LAW!

This document is copyrighted by the EIA and may not be reproduced without permission.

Organizations may obtain permission to reproduce a limited number of copies through entering into a license agreement. For information, contact:

Global Engineering Documents
15 Inverness Way East
Englewood, CO 80112-5704 or call
U.S.A. and Canada 1-800-854-7179, International (303) 397-7956

Exhibit 15 Page 207

SONY0005724

Contents

Exhibit 15 Page 208

Exhibit 15 Page 209

iii

Exhibit 15 Page 210

SONY0005727

Figures

Tables

Exhibit 15 Page 211

SONY0005728

v

Exhibit 15 Page 212

SONY0005729

(This page intentionally left blank.)

vi

Exhibit 15 Page 213

SONY0005730

FOREWORD

This document specifies the standards for Closed Captioning in Digital Television (DTV) technology. This standard was developed under the auspices of the Consumer Electronics Association (CEA, formerly CEMA) Technology & Standards R4.3 Television Data Systems Subcommittee in parallel with the U.S. Advanced Television Systems Committee's (ATSC) and the Advanced Television Grand Alliance's definition, design, and development of the audio, video and ancillary data processing standard for Advanced Television. The DTV standard developed by the Grand Alliance and other industry members is represented in ATSC A/53, and the informative document, ATSC A/54.

Users of this standard should note that, at some future point, it is expected that provisions necessary to accommodate NTSC line 21 data transmission will be established.

Exhibit 15 Page 214

(This page intentionally left blank.)

Exhibit 15 Page 215

SONY0005732

DIGITAL TELEVISION (DTV) CLOSED CAPTIONING

1 Scope

This document is intended as a definition of DTV Closed Captioning (DTVCC) and provides specifications and guidelines for caption service providers, DTVCC decoder and encoder manufacturers, DTV receiver manufacturers, and DTV signal processing equipment manufacturers. This specification includes the following:

- a description of the transport method of DTVCC data in the DTV signal
- a description of DTVCC specific data packets and structures
- a specification of how DTVCC information is to be processed
- a list of minimum implementation recommendations for DTVCC receiver manufacturers
- a set of recommended practices for DTV encoder and decoder manufacturers

The use of the term "DTV" (Digital Television) throughout is intended to include, and apply to, HDTV (High Definition Digital Television) and SDTV (Standard Digital Television) which use the digital data stream specified in ATSC A/53 and related ATSC A/54.

1.1 Overview

DTV Closed Captioning is a migration of the closed-captioning concepts and capabilities developed in the 1970's for NTSC television video signals to the high-definition television environment defined by the ATV (Advanced Television) Grand Alliance and standardized by the Advanced Television Systems Committee (ATSC). This new television environment provides for larger screens and higher screen resolutions, and higher data rates for transmission of closed-captioning data.

NTSC Closed Captioning consists of an analog waveform inserted into Line 21 of the NTSC Vertical Blanking Interval (VBI). This waveform provides a transport channel which can deliver 2 bytes of data on every field of video. This translates to 120 bytes per second (Bps), or 960 bits per second (bps). In contrast, DTV Closed Captioning is transported as a logical data channel in the DTV digital bit stream. Of the DTV bitstream bit rate (which is 19.4 Mbps for terrestrial broadcast, and 38.4 Mbps for cable), DTV-specific closed captioning is allocated 9600 bps. This increased capacity opens the possibilities for the simultaneous transmission of captions in multiple languages and at multiple reading levels.

The DTV standard accommodates a variety of increased vertical and horizontal screen resolutions (e.g., 480 x 704, 720 x 1280 and 1080 x 1920), versus the single 525 vertical scan line format for NTSC. These added resolutions provide for more defined representations of character fonts and other on-screen objects.

The heart of DTVCC caption display is the caption "window" which is identical to the *window* concept found in all computer Graphical User Interfaces (GUI). Windows are placed within the DTV screen, and caption text is placed within windows. Windows and text have a variety of color, size and other attributes.

This document describes the above issues in a reverse-hierarchical (i.e., low-to-high level) fashion. It follows an "Open Systems Interconnect (OSI) Reference Model"-type protocol stack for layered protocols. DTVCC consists of 5 protocol layers: the Transport Layer, the Packet Layer, the Service Layer, the Coding Layer, and the Interpretation Layer. The discussion of the first 3 layers is a detailed presentation of data transport and organization issues. The discussion of the last 2 layers provides a more informative presentation of the unique aspects of closed captioning. Some readers may wish to start with these last 2 layers first, beginning in section 7.

Throughout EIA-708-B, in concert with ATSC A/53, the lowest numbered bit in a multibit numbered value is considered to be least significant (uimsbf).

2 Normative References

The following references contain provisions, which, through reference in this text, constitute normative provisions of this standard. At the time of publication, the edition indicated was valid. All standards are subject to revision, and parties to agreements based on this standard are encouraged to investigate the possibility of applying the most recent edition of the standard indicated below.

1

2.1 Normative Reference List
ANSI X3.4, Information Systems - Coded Character Sets - 7-bit American National Standard Code (1986 (R1997))

ANSI X3.41, Code Extension Techniques for Use with the 7-Byte Coded Character Set of ASCII (1990)

ATSC A/53, ATSC Digital Television Standard (1995)

ATSC A/65, Program and System Information Protocol for Terrestrial Broadcast and Cable (1997)

EIA-608-A, Line 21 Data Service (1999)

ISO/IEC 2022, Information technology - Character code structure and extension techniques (1994)

ISO/IEC 8859-1, Information technology - 8-bit single-byte coded graphic character sets - Part 1: Latin alphabet No. 1 (1998)

ISO/IEC 13818-1, Information technology - Generic coding of moving pictures and associated audio information: Systems (1997)

SMPTE Standard 12M, Television, Audio and File – Time and Control Code (1999)

TIA/EIA-574, 9 Position Non-Synchronous Interface between Data Terminal Equipment and Data Circuit-Terminating Equipment Employing Serial Binary Data Interchange (1998)

2.2 Normative Reference Acquisition
ANSI or EIA or TIA/EIA Standards:
- Global Engineering Documents, World Headquarters, 15 Inverness Way East, Englewood, CO USA 80112-5776; Phone 800-854-7179; Fax 303-397-2740; Internet http://global.ihs.com; Email global@ihs.com

ATSC Standards:
- Advanced Television Systems Committee (ATSC), 1750 K Street N.W., Suite 1200, Washington, DC 20006; Phone 202-828-3130; Fax 202-828-3131; Internet http://www.atsc.org/stan&rps.html

ISO/IEC Standards:
- Global Engineering Documents, World Headquarters, 15 Inverness Way East, Englewood, CO USA 80112-5776; Phone 800-854-7179; Fax 303-397-2740; Internet http://global.ihs.com; Email global@ihs.com
- IEC Central Office, 3, rue de Varembe, PO Box 131, CH-1211 Geneva 20, Switzerland; Phone +41 22 919 02 11; Fax +41 22 919 03 00; Internet http://www.iec.ch; Email pubinfor@iec.ch
- ISO Central Secretariat, 1, rue de Varembe, Case postale 56, CH-1211 Geneve 20, Switzerland; Phone +41 22 749 01 11; Fax +41 22 749 01 55; Internet http://www.iso.ch; Email mbinfo@iso.ch

SMPTE Standards:
- Society of Motion Picture & Television Engineers, 595 W. Hartsdale Ave., White Plains, NY 10607-1824 USA Phone: 914-761-1100 Fax: 914-761-3115, Email: eng@smpte.org; Web: http://www.smpte.org

3 Caption Channel Layered Protocol
A formal data communications channel protocol has been established for the DTVCC caption data channel. This formalization provides a framework for describing the caption communications hierarchy. Grouping the structures, concepts, and features of this environment into the following hierarchical layers aids in the understanding of the organizational aspects of the DTVCC system.

There are 5 layers in the Caption Channel data framework: Transport Layer, Packet Layer, Service Layer, Coding Layer, and Interpretation Layer (see Figure 1). These layers map to the top-most layers of the OSI Reference Model as shown in Table 1.

2

Exhibit 15 Page 217

SONY0005734

OSI Protocol Reference Model	DTVCC Protocol Model
Application	Interpretation Layer
Presentation	Coding Layer
Session	Service Layer
-	Packet Layer
Transport	Transport Layer
Network	-
Link	-
Physical	-

Table 1 DTVCC Protocol Stack

The DTVCC Transport Layer maps to the OSI Transport Layer. This layer marks where DTVCC data leave the DTV Video subsystem and are introduced to the DTV Closed-Caption decoder in the receiving equipment. Within the DTVCC decoder, DTVCC data are further processed up through the remaining layers of the DTVCC Protocol Model.

The DTVCC Packet Layer marks where DTVCC data enter the DTVCC decoder. This is a protocol data reassembly layer which buffers incremental bitstream data into a byte-aligned, multi-byte packet. There is no specific counterpart in the OSI model for this layer.

Processing of the Caption Channel packet data begins in the DTVCC Service Layer. Caption Channel packets are broken up into the encapsulated sub-blocks of data to be routed to the separate caption service processing routines within the decoder. Services define separate caption data streams. TV viewers may choose to view the processed data for one or more services at a time. For example, a caption channel may contain an English language service and a Spanish language service.

The DTVCC Coding Layer breaks out the individual caption commands and caption text sequences from the service data blocks. The DTVCC Coding Layer maps to the OSI Presentation Layer.

The Interpretation Layer processes the caption elements presented by the DTVCC Coding Layer. The Interpretation Layer maps to the OSI Application Layer.

This layered framework for DTVCC is further detailed in the remaining sections of EIA-708-B.

Exhibit 15 Page 218

SONY0005735

DTVCC Interpretation Layer

Caption Data & Commands

DTVCC Coding Layer

DTVCC Service Data

DTVCC Service Layer

DTVCC Service Blocks

DTVCC Packet Layer

DTVCC Data Packet

DTVCC Transport Layer

DTV Signal Data
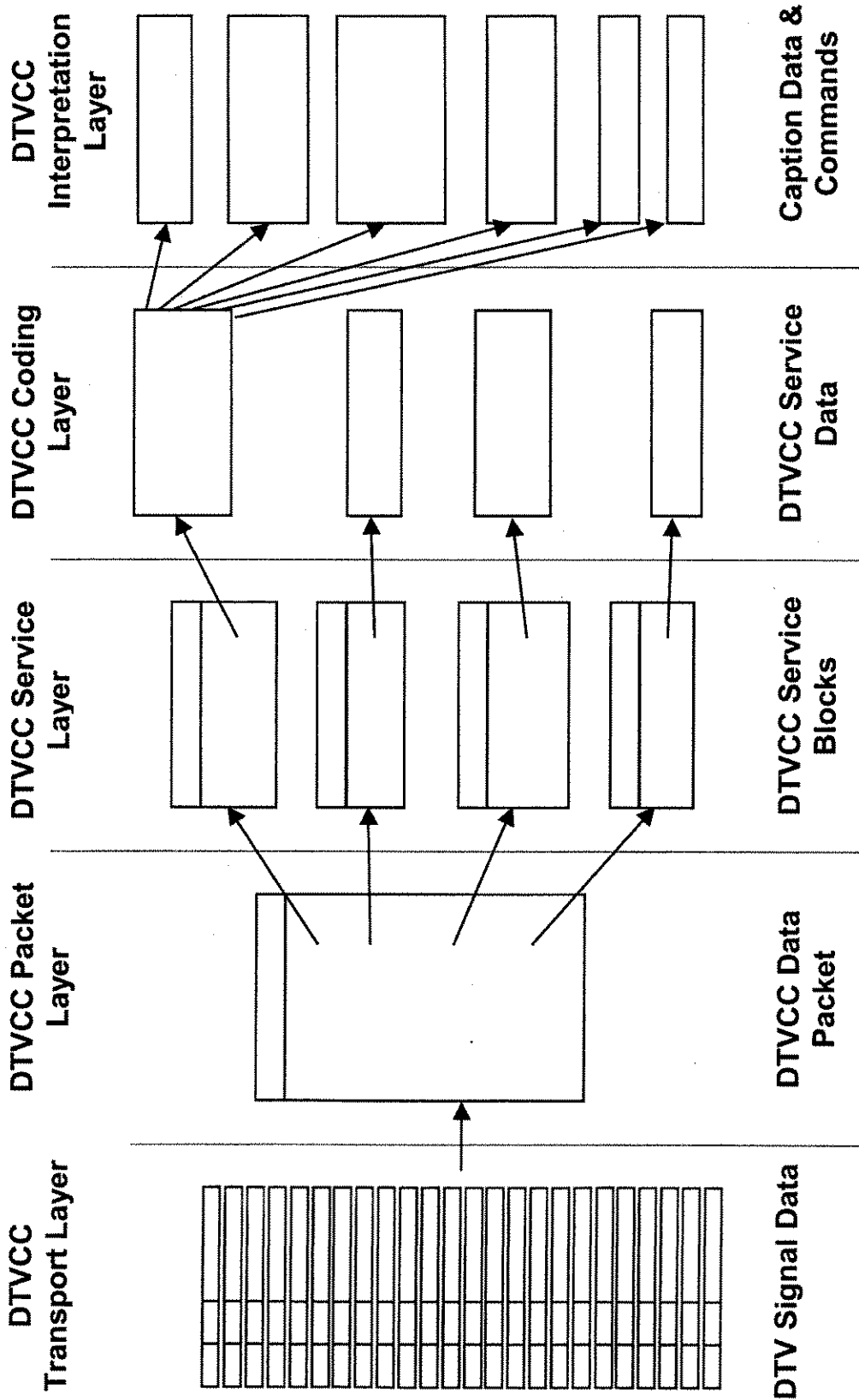
Figure 1 DTV Closed-Captioning Protocol Model

4

Exhibit 15 Page 219

SONY0005736

4 DTVCC Transport Layer

The transport of the Caption Channel is defined in ATSC A/53 and ISO/IEC 13818. The DTVCC Transport Layer consists of the mechanisms for transporting caption data from the encoder at the caption-encoding head-end to the decoding hardware in the TV receiver. DTVCC related data, when present, is transported in three separate portions of the DTV stream: the Picture User Data, the Program Mapping Table (PMT) and the Event Information Table (EIT). DTVCC Service Data (caption text, window commands, etc) are carried as MPEG-2 Picture User Data, and the DTVCC Caption Channel Service Directory is carried as descriptor information in the PMT and, when present, the EIT.

The DTV video bitstream, the PMT and the EIT are multiplexed with the other audio, data, control and synchronization bit streams comprising the DTV system signal as depicted in Figure 2.
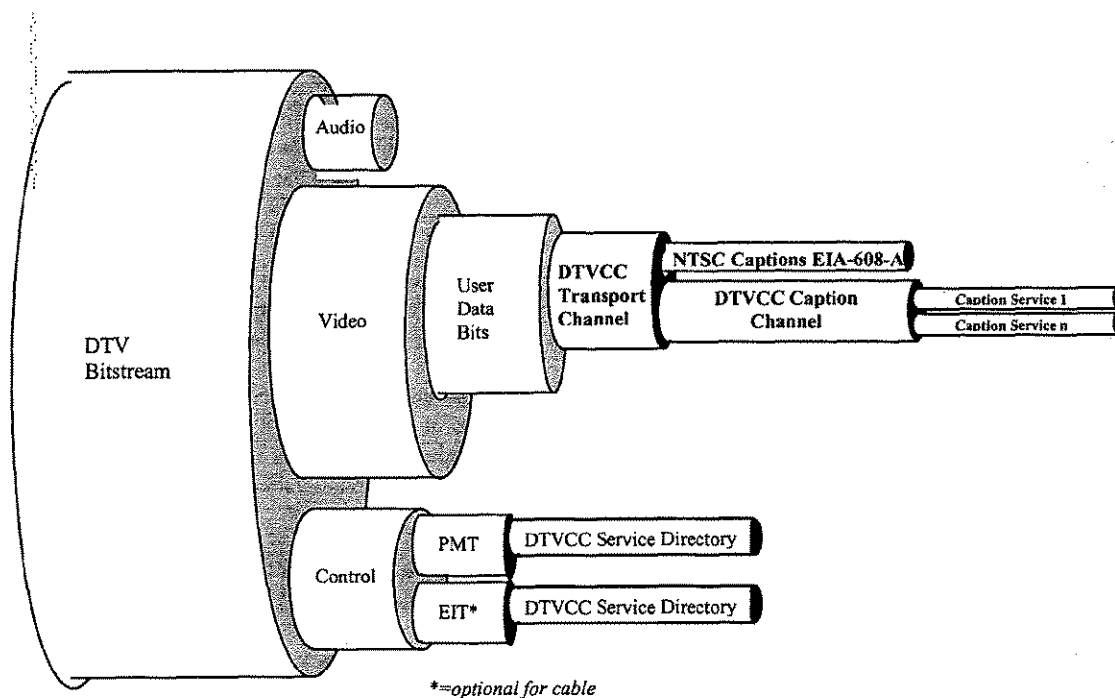
Figure 2 DTVCC Caption Data in the DTV Bitstream

Exhibit 15 Page 220

SONY0005737

4.1 9,600 Bits Per Second DTVCC Transport Channel

The DTVCC design allows for caption data to be transmitted at various data rates within a DTV signal or any similar MPEG-2 variation. However, for DTV and DTVCC specific (i.e., ATSC A/53) caption encoding, the DTV Closed-Captioning channel is a continuous 9600 bps stream allocated from the DTV signal capacity. In order to provide this continuous stream and to ensure that specific caption data will reach receivers as intended (in relationship to the audio and video), this data channel is allocated on a frame-by-frame basis such that 1200 bytes of data are transported per second. For example, for an interleaved DTV video signal with a 60 Hz frame rate, 20 bytes of DTVCC caption channel data are allocated in each frame.

It is very important to note that the allocation for NTSC closed-caption encoding within the DTVCC Transport Channel (i.e., Picture User Data) is included in the DTVCC Caption Channel bandwidth. That is, the total DTVCC Transport Channel (consisting of the DTVCC Caption Channel and the NTSC Caption Channel) within the Picture User Data is 9600 bps. On average, NTSC (EIA-608-A) captions are allocated 960 bps, and DTVCC captions (EIA-708-B) are allocated 8640 bps.

4.2 Pre-Allocated Bandwidth

The DTVCC Transport Channel is a fixed, pre-allocated stream which exists in all DTV-system bit streams, even though captions may not be present. This ever-present bandwidth (which includes NTSC and DTVCC caption data) allows encoders to easily insert caption data into the DTV bit-stream at the point of origin and at multiple down-stream encoding points without having to perform complex picture data processing and bandwidth re-allocation. In addition, it allows for easier decoding of the closed-caption data by the receiver since the data will always be at known locations.

4.3 NTSC Caption Data

The consideration for NTSC captioning (EIA-608-A) exists to facilitate the transcoding of the DTV video to NTSC video, and to preserve all line 21 data during the transcode process. The NTSC closed-captioning allocation is detailed further in this section. Section 9.23 provides a further discussion of the transcoding issue.

The NTSC closed caption data bytes are not embedded within the DTV closed-caption protocol stack. That is, they are not passed onto the DTVCC Packet Layer; rather, they are extracted at the Transport Layer and routed to a separate NTSC caption decoder (if present). This allows for simpler closed-caption decoder implementations (e.g., DTV-to-NTSC settop transcoders) since the entire DTVCC Caption Channel data stream does not have to be parsed to find a few bytes of NTSC data.

This is especially true for devices (e.g., set-top boxes) which are to convert DTV signals into NTSC to accommodate NTSC television receivers. These devices ignore the DTVCC closed-caption data in the DTV Picture User Data bits and transcode the NTSC closed-caption data into the NTSC Line-21 data stream. This allows for easier insertion of NTSC closed-caption data by the encoder since the NTSC data bytes do not have to be buried within the DTVCC Coding Layer.

4.4 DTV/MPEG-2 Picture User Data (User Bits)

While Video User Bits may be inserted at any of three levels (i.e., Sequence level, the Group of Pictures (GOP) level, and the Picture Data level) within the video bitstream sequence, ATSC A/53 specifies that the DTVCC Transport Channel is only encoded at the Picture Data level (i.e., user data level 2).

The following is a technical discussion of the DTV Picture User Data format. This discussion is intended for the technical audience, such as encoder and decoder manufacturers. The bitstream description syntax convention used in the two referenced documents is also used here to describe the DTV user data. Please refer to these documents for a normative explanation of this syntax.

4.4.1 Picture User Data Bitstream Construct

The extension_and_user_data(2) and user_data() constructs defined in ATSC A/53 carry the DTVCC data within the MPEG Video Picture User Data Level. This section augments the definition of the *cc_type* and *cc_valid* elements in the user_data(2) construct.

cc_valid - This flag indicates whether the following two closed-caption data bytes are valid. If *cc_valid* = 1, then the following two bytes of closed-caption data are valid. Otherwise, the two closed caption bytes are invalid (i.e., null

6

filled). For NTSC Line 21, Field 1 and Field 2 captions, if *cc_valid* = 0 then the run-in clock and start bit should not be generated for the NTSC data stream when transcoding from DTV to NTSC.

cc_type - Denotes the type of the following two bytes of closed caption data as indicated in Table 2.

cc_type	Contents
00	NTSC line 21 field 1 closed captions
01	NTSC line 21 field 2 closed captions
10	DTVCC Channel Packet Data
11	DTVCC Channel Packet Start

Table 2 Closed-Caption Type (cc_type) Coding

DTV Closed-Caption data is transmitted (encoded) in variable-rate, variable-sized DTVCC Caption Channel Packets (see Section 5). To allow for simple extraction and insertion of these packets without the need to fully parse their contents, the first byte-pair of each DTVCC Caption Channel Packet shall be marked using the *cc_type* = 11 syntax flag (i.e., DTVCC Caption Channel Packet Start). The remaining byte-pairs of the DTVCC packet shall be marked with the *cc_type* = 10 flag (i.e., DTVCC Caption Channel Packet Data).

A DTVCC Caption Channel Packet may continue from one picture user_data extension to the next. The end of the packet is indicated by either: (1) receipt of the header (*cc_valid* = 1, *cc_type* = 11) of the next DTVCC packet, or (2) receipt of a byte-pair where *cc_valid* = 0 and *cc_type* = 10 or *cc_type* = 11.

> NOTE--In the DTV video compression standard, coded pictures are transmitted in a different order ("Transport" order) than they are displayed ("Display" order). Since the DTVCC captioning data extensions are part of the video coded picture constructs and follow the MPEG Video coded-picture reordering process, the order in which the captioning data extensions are transmitted is not the same order in which they will be processed by the DTVCC Decoder. The captioning data extensions must be reordered (by the "MPEG Video Decoder") along with the pictures to which they correspond prior to the DTVCC Caption Channel Packet location and extraction method described previously. Once these captioning data are reordered, they will be ready for processing by the DTVCC Decoder.

4.4.2 Frame Rates

As stated above, the value of *cc_count* will vary depending on the frame rate. In any given video service, the frame rate may change (e.g, to "film-mode" during extended still picture periods). The *user_data()* construct shall be transmitted a number of times per second, and *cc_count* shall be adjusted for each such construct so that the 9600 bps DTV Transport Channel throughput is sustained no matter what the instantaneous frame rate is.

The frame rate for a DTV video stream is dependent upon a combination of the values for the following MPEG-2 parameters (i.e., video bitstream elements): *frame_rate_code, progressive_sequence, top_field_first, repeat_first_field* and *picture_structure*. These parameters are defined in ISO/IEC 13818. When defining the frame rate, these parameters also govern the *user_data()* construct transmit rate and the value of *cc_count*.

Table 3 shows the values derived for the *user_data(2)* transmit rate, and the *cc_count* values per each frame such that the DTVCC Transport Channel data rate of 9,600 bps is maintained.

Table 3 also shows the relationship between the number of DTV data bytes vs. NTSC bytes contained in the DTVCC user data.

NTSC captioning information shall not exceed the data rate of an NTSC display (30/1.001 * 2 bytes per second). The number of cc_count of each NTSC cc_type shall be no greater than 30 /1.001 per second while the stream is active. This data rate may be maintained by averaging the number of each cc_type over up to 1 001/30 seconds.

This requirement must be met across splices. For example, when two sequences with frame_rate_value of either 30 or 60 are spliced, the splicer could delete NTSC caption information from either the 1/30 second of the ending sequence or the first 1/30 second of the new sequence.

7

NTSC captioning information should maintain the order of an NTSC display. If both cc_type 00 and cc_type 01 data are present in the picture user data constructs of a sequence, they should alternate when the pictures are restored to display order.

For video sequences with progressive_sequence ==0 for pictures with top_field_first==0, cc_type 00 shall precede cc_type 01.

For video sequences with progressive_sequence ==0 for pictures with top_field_first==1, cc_type 01 shall precede cc_type 00.

For picture_structure of Top Field, cc_type 00 shall not be present.

For picture_structure of Bottom Field, cc_type 01 shall not be present.

For video sequences with progressive_sequence == 1, the ordering of cc_type 00 and cc_type 01 is arbitrary. Either 4 or 6 NTSC cc_data bytes may be included in each *user_data*() structure, subject to the data rate restriction above. DTV cc_data bytes may also be present to produce a total data rate of 9120 bps, subject to the constraint on cc_count.

When the picture display time is extended from the nominal value of frame_rate stated in the video sequence header through the repeat mechanisms in the picture_coding_extension, the values of cc_count and number of NTSC cc_data_bytes in user_data() shall be modified. Values for cc_count and cc_data_bytes are shown in Table 3.

Frame_rate_value	picture_structure	progressive_frame	repeat_first_field	top_field_first	cc_count	NTSC cc_data_bytes	DTV cc_data_bytes
60 or 59.94	11	1	0	0	10	2	18
60 or 59.94	11	1	1	0	20	4	36
60 or 59.94	11	1	1	1	30	6	54
30 or 29.97	01 or 10	0	0	0	10	2	18
30 or 29.97	11	x	0	x	20	4	36
30 or 29.97 (note)	11	1	1	x	30	6	54
24 or 23.97	11	1	0 (required)	0 (required)	25	4 or 6	44 or 46
(x = may be 0 or 1) Note: For sequences with frame_rate_value = 30, or 29.97, progressive_frame ==1 and repeat_first_field==1 implies progressive_sequence==0.							

Table 3 DTVCC Transport Channel Transmit Rate Parameters

For ease of inserting and retrieving the NTSC closed captions, NTSC Line 21, Field 1 and Field 2 captions must always be placed in the User Data stream before any DTVCC caption data. The NTSC closed-caption data order (Field 1 vs. Field 2) will depend on the field display order. With this scheme, once the decoder sees cc_type = 10 or cc_type = 11, indicating DTVCC caption data, it will no longer continue searching the User Data for NTSC captions. All NTSC closed-caption data will follow the format and protocol as specified in EIA-608-A.

In the case of a 60 Hz frame rate, 2 bytes of NTSC captions are allocated per user_data extension. Two (2) bytes of Field 1 NTSC captions are to be encoded every other frame, with 2 bytes of Field 2 NTSC captions encoded in the intervening frames. In all other cases (i.e., lower frame rates), Field 1 and Field 2 captions exist within the same user_data extension.

Exhibit 15 Page 223

SONY0005740

4.4.3 Typical Video Signals

Table 4 illustrates an example of the arrangement of the closed-caption data stream in the Picture User Data for a typical 30 frames per second video signal frame where *cc_count* = 20. A Caption Channel Packet starts immediately after the NTSC caption data.

	Marker_bits	cc_valid	cc_type	cc_data1	cc_data2
1	1111 1	1	00	NTSC Line 21 field 1	NTSC Line 21 field 1
2	1111 1	1	01	NTSC Line 21 field 2	NTSC Line 21 field 2
3	1111 1	1	11	byte #1: DTVCC Pkt Header	byte #2: DTVCC Pkt Data
4	1111 1	1	10	byte #3: DTVCC Pkt Data	byte #4: DTVCC Pkt Data
•	•	•	•	•	•
•	•	•	•	•	•
•	•	•	•	•	•
20	1111 1	1	10	byte #35: DTVCC Pkt Data	byte #36: DTVCC Pkt Data

Table 4 Aligned User Data and DTVCC Channel Packet Example

Table 5 shows the User Data frame encoding of unaligned Caption Channel Packets. The end of a previous Caption Channel Packet (that was 128 bytes in length) is in the first 4 bytes of the User Data after the NTSC caption data. The start of the first 32 bytes of a new Caption Channel Packet fills the remainder of the 40 bytes of the User Data for a 30 frames per second video signal.

	Marker_bits	cc_valid	cc_type	cc_data1	cc_data2
1	1111 1	1	00	NTSC Line 21 field 1	NTSC Line 21 field 1
2	1111 1	1	01	NTSC Line 21 field 2	NTSC Line 21 field 2
3	1111 1	1	10	byte #125: DTV Pkt Data	byte #126: DTV Pkt Data
4	1111 1	1	10	byte #127: DTV Pkt Data	byte #128: DTV Pkt Data
5	1111 1	1	11	byte #1: DTVCC Pkt Header	byte #2: DTVCC Pkt Data
6	1111 1	1	10	byte #3: DTVCC Pkt Data	byte #4: DTVCC Pkt Data
7	1111 1	1	10	byte #5: DTVCC Pkt Data	byte #6: DTVCC Pkt Data
8	1111 1	1	10	byte #7: DTVCC Pkt Data	byte #8: DTVCC Pkt Data
•	•	•	•	•	•
•	•	•	•	•	•
•	•	•	•	•	•
20	1111 1	1	10	byte #31: DTVCC Pkt Data	byte #32: DTVCC Pkt Data

Table 5 Unaligned User Data and DTVCC Channel Packet Example

The *marker_bits, cc_type* and *cc_valid* fields DO NOT use any part of the 9,600 bps bandwidth allocated for closed-caption data. Therefore, 9,600 bps will be maintained for the transport of closed-caption information, including NTSC and DTVCC captions, within the DTV stream.

4.4.4 Latency

As previously discussed in Section 4.4.1, the DTV picture data are transmitted in "Transport Order". This transmission path has an inherent latency associated with it, in that there can be a significant delay from the time the picture data enter a DTV encoder to the time these same data exit a DTV decoder and are available for display processing. The DTVCC data experience this same delay since they are transmitted within the Picture User Data bits, even though the actual DTVCC data are introduced to the encoder in "Display Order', as is the video. ATSC A/53 specifies that this latency may not exceed 0.5 seconds.

This requirement imposes a constraint on the *vbv_delay* parameter of the MPEG-2 Picture Header bitstream construct. This value is the Video Buffering Verifier (VBV) Delay. The VBV is a hypothetical decoder that is

9

Exhibit 15 Page 224

SONY0005741

conceptually connected to the output of the DTV encoder. Its purpose is to provide a constraint on the variability of the data that an encoder or editing process may produce.

vbv_delay is a 16-bit unsigned integer in the Picture Header. For constant bit rate operation, *vbv_delay* is used to set the initial occupancy of the decoder's video buffer at the start of play so that the decoder's buffer does not overflow or underflow. *vbv_delay* measures the time needed to fill the VBV buffer from an initially empty state at the target bit rate, R, to the correct level immediately before the current picture is removed from the buffer.

The value of *vbv_delay* is the number of periods of the 90 kHz system clock that the VBV shall wait after receiving the final byte of the picture start code. For a 0.5 second delay, the *vbv_delay* value is 45,000 (i.e., 45,000 cycles / 90,000 cycles per second = 0.5 seconds). If *vbv_delay* <= 45,000 for a picture start code, it means that the closed captions in that picture data will experience a delay of less than 0.5 second in the decoder buffer.

4.5 Caption Channel Service Directory in the PMT and EIT

The Caption Channel Service Directory describes the types and attributes of the Caption Services (see Section 6) encoded in the Picture User Data bits. The Caption Channel Service Directory is contained in the caption_service_descriptor().The caption_service_descriptor() is defined by ATSC A/65, and is carried in the PMT and, when present, the EIT of the MPEG-2 transport stream. Reference ATSC A/65 for a complete description of the caption_service_descriptor.

4.5.1 PMT and EIT Constraints

The following constraints shall be placed on the use of the caption service descriptor:

1) One caption service descriptor shall be present in the PMT and, when present, the EIT to describe each caption service present within the video Picture User Data.

2) There shall be no more than 16 simultaneous caption services present.

4.5.2 PMT Bandwidth Requirements

For the case where the PMT is repeated at the minimum rate of once per 400 msec, the bandwidth needed to deliver a full set of 16 caption service descriptions is 1.98 kbps (i.e.,3 bytes overhead + (6 bytes/description * 16 descriptions) = 99 bytes. 99*8=792 bits. 792 * 2.5 reps/second = 1980 bps). Packetization overhead is not included in this rate.

4.5.3 EIT Bandwidth Requirements

According to ATSC A/65, the recommended cycle time for the currently applicable EIT is 500 msec. Using the same assumptions as the previous paragraph (full set of services for a total of 99 bytes), the bandwidth required is 1.584 kbps.

4.5.4 Decoder Processing of the PMT, EIT and User Data

Even though the Caption Service Descriptors are encoded in the PMT and, when present, the EIT, the decoder only needs to acquire this information from one location. For terrestrial broadcasts, decoders may acquire service descriptors from the EIT only. The EIT carries all relevant event information for a particular program (program titles, duration, ratings, caption directory, etc.). Acquiring all program information from a single, concise table facilitates fast tuning.

In other transmission environments, it may be preferable to use the PMT for acquisition of the Caption Service Descriptor. See Annex A (informative).

5 DTVCC Packet Layer

The DTVCC Caption Channel data within the DTVCC Transport Channel are framed within data "packets" prior to encoding. The DTVCC Caption Channel Packet is totally defined within EIA-708-B. Error correction, error detection, compression, and other low-level transport overhead issues are handled in the transmission DTV layers, and thus fall into the purview of the ATSC and MPEG-2 standards.

The DTVCC Packet Layer is defined by a Caption Channel Packet of **n** bytes of closed captioning data, where **n** <= 128 and **n** is an even number. Packets are coded in the user_data extensions of the coded pictures in the DTV stream. The beginning and ending of each DTVCC packet in the User Data is indicated in the syntax defined in Section4.4.1.

Exhibit 15 Page 225

SONY0005742

The Caption Channel Packet consists of a 1 byte header and **n**-1 bytes of packet data, where **n** is the total packet size. See Figure 3. The Caption Channel Packet Header contains the packet_size code and a sequence_number. The packet_size_code is in the lower 6 bits (b0 - b5) of the header; its value is "0" when the packet size (**n**) is 128 bytes and "**n**/2" when the packet size is less than or equal to 126 bytes.

Sequence_number is a 2-bit (b6 - b7) rolling sequence number (0 - 3) which is used by receivers to determine lost Caption Channel Packets. When a lost packet is detected, any partially accumulated data from the previously received packet is to be discarded, and the processing associated with the Reset comand shuld be performed for each existing service. See section 8.9.5. Decoding shall resume with the first Service Block encountered in the new packet.

b7	b6	b5	b4	b3	b2	b1	b0	
Seq. No.		Packet Size (n/2)						Packet Header
Caption Channel Data Byte 1								
Caption Channel Data Byte 2								Packet Data
. . .								
Caption Channel Data Byte n-1								
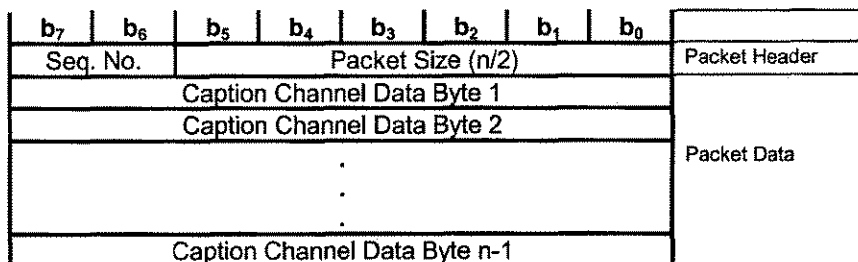
Figure 3 Caption Channel Packet

The syntax for the caption_channel_packet is shown in Figure 4.

```
caption_channel_packet {
        /* packet header   */
        sequence_number: 2 bits
        packet_size: 6 bits


        /* packet data sequence */
        /*if (packet_size_code == 0)*/
                /*packet_data_size = 127 */
        /*else*/
                /*packet_data_size = (packet_size_code * 2) –1 */
        for (i = 0; i < packet_data_size; i++)
        {
                Packet_data[i]    : 8 bits;
        }
}
```

Figure 4 Caption Channel Packet Syntax

6 Caption Service Layer
The DTVCC Caption Channel is divided into a set of logical sub-channels; i.e., "services". Service data are inserted in the Caption Channel when required and where it fits in the caption channel data stream (i.e., Time Division Multiplexed).

The Service Layer defines the headers for the caption data channel service numbers, service types, and service attributes. Receivers use the information in this layer to route the caption packets to the appropriate internal processing modules.

6.1 Services
Caption Channel services are virtual sub-channels in the Caption Channel stream. There are 6 standard services and up to 57 additional extended services allowing for 63 total services.

11

Exhibit 15 Page 226

SONY0005743

Service #1 is designated as the Primary Caption Service. This service contains the verbatim, or near-verbatim captions for the primary language being spoken in the accompanying program audio.

Service #2 is designated as the Secondary Language Service. This service contains captions in a secondary language which are translations of the captions in the Primary Caption Service.

The other service sub-channels are not pre-assigned. It is up to the discretion of the individual caption provider to utilize the remaining service channels.

6.2 Caption Channel Service Blocks

Caption Channel Service Blocks provide the structures for the asynchronous time division multiplexing of service data within the DTV Closed-Caption Channel. Caption providers and encoding equipment algorithms govern the frequency, priority, and bandwidth consumption for each individual service by these service blocks, as needed.

A Caption Channel Service Block consists of a Service Block Header followed by from 1 to 31 bytes of service data. See Figure 5. As described in 6.2.1, a Service Block Header may consist of 1 or 2 bytes.

Service Block Header (1 or 2 bytes)
Service Block Data Byte 1
· · ·
Service Block Data Byte m (m <= 31)
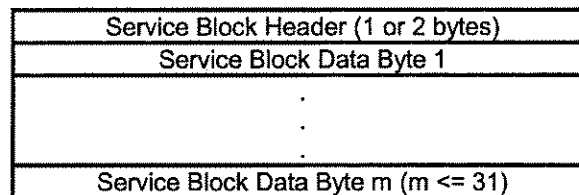
Figure 5 Service Block

NOTE--A Service Block cannot span a Caption Channel Packet. Service Blocks must begin and end within the same Caption Channel Packet.

The syntax for the Service Block is shown in Figure 6.

```
service_block {
        /* service block header */
        block_size        : 5 bits   /*    >= 0 && <= 31       */
        service_number   : 3 bits

        /* extended service block header extension */
        if (service_number == b'111' && block_size != 0)
        {
                extended_service_number : 6 bits
                null_fill                        : 2 bits
        }

        /* block data sequence */
        if (service_number != 0)
        {
                for (i = 0; i < block_size; i++)
                {
                        Block_data[i]     : 8 bits
                }
        }
}
```

Figure 6 Service Block Syntax

12

Exhibit 15 Page 227

SONY0005744

6.2.1 Standard Service Block Header

The one-byte Standard Service Block Header consists of 2 parts: the Service Number (sn), and the Service Block Size (bs). See Figure 7. The Service Number (sn_0 - sn_2) is defined in the 3 high-order bits of the block header byte, and the Block Size (bs_0 - bs_4) is defined in the 5 low-order bits of the block header byte.

"Standard" service numbers range from 1 to 6 (service #0 is reserved), with Service #1 being reserved for the Primary Caption Service, and Service #2 reserved for the Secondary Language Service. A service number value of 7 in bits sn_0 - sn_2 indicates an "Extended" Service Block (Section 6.2.2).

The service Block Size value ranges from 1 to 31 and indicates the number of bytes following the header. A Standard Service Block Size of 0 is meaningless.

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
sn_2	sn_1	sn_0	bs_4	bs_3	bs_2	bs_1	bs_0

Figure 7 Standard Service Block Header

6.2.2 Extended Service Block Header

In the event that a Caption Channel requires more than 6 simultaneous services, a 2-byte Extended Service Block Header is used. See Figure 8. The first byte of this extended header has the same format as the Standard Service Block Header with the exception that the high order 3 bits are fixed to a value of all 1's. This standard service number value of "7" identifies the Extended Service Block and indicates that the 2nd byte of the header contains an extended service number (sn_0 - sn_5) whose value may range from 7 to 63. Extended service numbers less than 7 are illegal (since they can already be specified with the Standard Service Block Header).

The Block Size (bs_0 - bs_4) value in the lower 5 bits of the first byte of the Extended Service Block Header is the number of bytes following the Extended Service Block Header.

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
1	1	1	bs_4	bs_3	bs_2	bs_1	bs_0
0	0	sn_5	sn_4	sn_3	sn_2	sn_1	sn_0

Figure 8 Extended Service Block Header

6.2.3 Null Service Block Header

A Null Service Block Header must be inserted as the last Service Block in the Caption Channel Packet, space permitting. This header type indicates that there are no more Service Blocks in the packet for the DTVCC decoding hardware to process. See Figure 9. Encoding equipment should null fill (i.e., zero out) the Caption Channel Packet buffer before Service Blocks are inserted, thus insuring that a null header is always present in non-full Caption Channel Packets.

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
0	0	0	0	0	0	0	0

Figure 9 Null Service Block Header

6.2.4 Service Block Data

Up to 31 bytes of Service Data follow the Service Block Header within the Service Block. The contents of the data portion of the service block consists of caption data coding and interpretation (Sections 7 and 8). Service Block Data are separated from the Service Block and are routed to the appropriate service processors in the DTV receiver decoder. This separation process creates the individual byte streams for each of the services which are handed off to the DTVCC Coding Layer.

6.2.5 Service Blocks in Caption Channel Packet

Service Blocks are time division multiplexed and inserted sequentially in the Caption Channel Packet path (Section 4.5).

13

Exhibit 15 Page 228

SONY0005745

Service Blocks may not cross Caption Channel Packet boundaries; i.e., if a service requires more data than the current packet allows, then the service block is truncated to fit the current packet, and a new Service Block with a new Service Block Header and the remaining data bytes are placed at the start of, or elsewhere, within the next packet.

Figure 10 shows an example of a Caption Channel Data Packet with 3 Standard Service Blocks and 1 Extended Service Block. The Caption Channel Packet size is 20 bytes and the Packet Sequence Number is 2.

									Packet Byte
Pkt Size: 20/2, Seq# 2	1	0	0	0	1	0	1	0	0 (Packet Header)
SN: 1, BS: 3	0	0	1	0	0	0	1	1	1 (Service 1)
									2
									3
									4
SN: 6, BS: 4	1	1	0	0	0	1	0	0	5 (Service 6)
									6
									7
									8
									9
ESN: 21,	1	1	1	0	1	0	0	0	10 (Ext. Service 21)
BS: 8	0	0	0	1	0	1	0	1	11
									12
									13
									14
									15
									16
									17
									18
									19
Null Service Header	0	0	0	0	0	0	0	0	20

Figure 10 Service Blocks in Caption Channel Packets (Example)

7 DTVCC Coding Layer - Caption Data Services (Services 1 - 63)

The DTVCC Coding Layer describes how data are coded for the caption channel services. The Caption Data Coding Layer defines the assignment of numerical codes for code space control, caption commands, and caption characters and symbols.

7.1 Code Space Organization

Consistent with ANSI X3.41 and ISO 2022, the 256 position code space is divided into four code groups: the CL, GL, CR and GR. See Table 6 and Table 7. Each group contains a standard code set and an extended code set.

- The CL group contains the 32 addressable codes from 00h to 1Fh. The C0 (a subset of the ASCII, ANSI X3.4 Miscellaneous Control Codes) and C2 (Extended Miscellaneous Control Codes) code sets are mapped to this space.
- The GL group contains the 96 addressable codes from 20h to 7Fh. The G0 (a slightly modified version of the ANSI X3.4 ASCII Printable Character set) and G2 (Extended Control Code Set 1) code sets are mapped to this space.
- The CR group contains the 32 addressable codes from 80h to 9Fh. The C1 (Caption Control Codes) and C3 (Extended Control Code Set 2) code sets are mapped to this space.
- The GR group contains the 96 addressable codes from A0h to FFh. The G1 (ISO 8859-1 Latin 1 Characters) and G3 (future character and icon expansion) code sets are mapped to this space.

14

Exhibit 15 Page 229

SONY0005746

b7-b4:	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
b3-b0																
0																
1																
2																
3																
4																
5																
6	**CL**		**GL**						**CR**		**GR**					
7	**(C0, C2)**		**(G0, G2)**						**(C1, C3)**		**(G1, G3)**					
8	**Set**		**Set**						**Set**		**Set**					
9																
A																
B																
C																
D																
E																
F																

Table 6 DTVCC Code Space Organization

Exhibit 15 Page 230

SONY0005747

EIA-708-B

b3-b0	C 0		G 0						C 1		G 1					
b7-b4	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	NUL	EXT1	SP	0	@	P	`	p	CW0	SPA	NBS	°	À	Ð	à	ð
1			!	1	A	Q	a	q	CW1	SPC	¡	±	Á	Ñ	á	ñ
2			"	2	B	R	b	r	CW2	SPL	¢	²	Â	Ò	â	ò
3	ETX		#	3	C	S	c	s	CW3		£	³	Ã	Ó	ã	ó
4			$	4	D	T	d	t	CW4		¤	´	Ä	Ô	ä	ô
5			%	5	E	U	e	u	CW5		¥	µ	Å	Õ	å	õ
6			&	6	F	V	f	v	CW6		¦	¶	Æ	Ö	æ	ö
7			'	7	G	W	g	w	CW7	SWA	§	·	Ç	×	ç	÷
8	BS	P16	(	8	H	X	h	x	CLW	DF0	¨	¸	È	Ø	è	ø
9			)	9	I	Y	i	y	DSW	DF1	©	¹	É	Ù	é	ù
A			*	:	J	Z	j	z	HDW	DF2	ª	º	Ê	Ú	ê	ú
B			+	;	K	[	k	{	TGW	DF3	«	»	Ë	Û	ë	û
C	FF		,	<	L	\	l			DLW	DF4	¬	¼	Ì	Ü	ì	ü
D	CR		-	=	M	]	m	}	DLY	DF5		½	Í	Ý	í	ý
E	HCR		.	>	N	^	n	~	DLC	DF6	®	¾	Î	Þ	î	þ
F			/	?	O	_	o	♪	RST	DF7		¿	Ï	ß	ï	ÿ
0			TSP	█							CC					
1			NBTSP	‛												
2				'												
3				"												
4				"												
5			...	•												
6								⅛								
7								⅜								
8								⅝								
9				TM				⅞								
A			Š	š				│								
B																
C			Œ	œ				└								
D				SM				—								
E																
F				Ÿ				⌐								

| C 2 | | G 2 | | C 3 | | G 3 | |

Table 7 DTVCC Code Set Mapping

NOTE--This section lists all required and optional caption codes. Refer to Section 9.4 for a list of required codes for a minimum DTVCC receiver implementation.

16

Exhibit 15 Page 231

7.2 Extending the Code Space

The characters in the extended code sets of the CL, CR, GL, and GR code groups are accessed using the **EXT1** code (10h) in the C0 code set (Miscellaneous Control Codes). Normally (i.e., without extending the code space) the base codes in the four code groups (CL, GL, CR, and GR) represent the characters, control codes, and commands in the C0, C1, G0, and G1 code sets. By prefixing the codes in the code groups with the **EXT1** code (and thus forming the start of a two-byte code), the symbols in the extended C2, C3, G2, and G3 code sets are referenced; that is, each character in these code sets require the transmission of two codes (i.e., **EXT1** plus a base code) in order to be referenced. **EXT1** must start every 2-byte extended code sequence.

When **EXT1** is not encountered, then reference to the base code sets (C0, C1, G0, and G1) is assumed. That is, **EXT1** is only active for the two-byte extended code sequence in which it exists.

For example, to generate the closed-caption symbol (CC) in the G3 code set, the following sequence must be specified: 10h, A0h (**EXT1, NBS**).

The code space may be extended further (for future use) to handle other character sets not included in the language sets listed in section 7.4.4. This extension capability anticipates code sets which require 16-bit character code addressing, such as characters to implement Chinese and Japanese. The **P16** code in the C0 Miscellaneous Control Code set is used for this purpose. When a decoder encounters the **P16** code, it uses the succeeding two bytes to address characters in a 16-bit code set.

7.3 Unused Codes

Any codes not specified in the following sections are reserved for future standardization. Decoders encountering any undefined codes should adhere to the size (e.g., 1 byte, 2 byte, 3 byte and variable length) characteristics of these codes, as indicated in the following sections. This assures that future extensions to the coding scheme can be ignored by decoders which do not support them.

7.4 Numerical Organization of Codes

The following subsections detail the mapping of the individual code sets (C0, C1, C2, C3, G0, G1, G2, and G3) of the DTVCC code space.

7.4.1 C0 Code Set - Miscellaneous Control Codes

The C0 code set contains the 32 addressable codes from 00h to 1Fh. See Table 8.

> Codes 00h through 0Fh are single byte codes.
> Codes 10h through 17h are two byte codes.
> Codes 18h through 1Fh are three byte codes.

The **NUL, BS, FF,** and **CR** codes are preserved from the standard ASCII control code set. The **ETX** code is also from the ASCII control code set, but it has a special use in that it is required at the end of a caption text segment to terminate the segment when the segment is not immediately followed by another caption command (see Section 8.10.4). The **EXT1** code is used to extend the DTVCC code space (see Section 7.2). The **P16** code is used as a further code space extension for 16-bit character sets.

Exhibit 15 Page 232

SONY0005749

b7-b4	C	0
b3-b0	0	1
0	NUL	EXT1
1		
2		
3	ETX	
4		
5		
6		
7		
8	BS	P16
9		
A		
B		
C	FF	
D	CR	
E	HCR	
F		

Table 8 C0 Code Set

7.4.2 C1 Code Set - Captioning Command Control Codes

The C1 code set contains the 32 addressable codes from 80 to 9F. See Table 9. This set contains the captioning command control codes (window creation commands, character attributes, etc.). The use of these codes and the DTVCC captioning commands are detailed in Section 8.10.

b7-b4	C	1
b3-b0	8	9
0	CW0	SPA
1	CW1	SPC
2	CW2	SPL
3	CW3	
4	CW4	
5	CW5	
6	CW6	
7	CW7	SWA
8	CLW	DF0
9	DSW	DF1
A	HDW	DF2
B	TGW	DF3
C	DLW	DF4
D	DLY	DF5
E	DLC	DF6
F	RST	DF7

Table 9 C1 Code Set

18

Exhibit 15 Page 233

SONY0005750

7.4.3 G0 Code Set - ASCII Printable Characters

The G0 code set contains the 96 addressable codes from 20h to 7Fh. See Table 10. This set provides the ANSI X3.4 ASCII printable characters with the substitution of the music note character for the ASCII **DEL** character.

b7-b4 → b3-b0 ↓	2	3	4	5	6	7
0	SP	0	@	P	`	p
1	!	1	A	Q	a	q
2	"	2	B	R	b	r
3	#	3	C	S	c	s
4	$	4	D	T	d	t
5	%	5	E	U	e	u
6	&	6	F	V	f	v
7	'	7	G	W	g	w
8	(	8	H	X	h	x
9	)	9	I	Y	i	y
A	*	:	J	Z	j	z
B	+	;	K	[	k	{
C	,	<	L	\	l	|
D	-	=	M	]	m	}
E	.	>	N	^	n	~
F	/	?	O	_	o	♪

Table 10 G0 Code Set

Exhibit 15 Page 234

SONY0005751

EIA-708-B

7.4.4 G1 Code Set - ISO 8859-1 LATIN-1 Character Set

The G1 code space contains the 96 addressable codes from A0h to FFh. See Table 11. This set consists of the ISO 8859-1 Latin-1 character set, also known as the Windows/ANSI character set. This set, when used in conjunction with the ASCII set, provides all the characters needed to encode text in Danish, Dutch, Faeroese, Finnish, French, German, Icelandic, Irish, Italian, Norwegian, Portuguese, Spanish and Swedish. Many other languages can be written with this set of letters, including Hawaiian, Indonesian/Malay, and Swahili. ISO 8859-1 also extends the ASCII set with additional miscellaneous punctuation and mathematical signs.

Code **NBS** (A0h) represents a non-breaking space. This code is to be used (instead of a space character) between words that should not be split when word-wrapping is in effect.

	G		**1**			
b7-b4 b3-b0	A	B	C	D	E	F
0	NBS	°	À	Ð	à	ð
1	¡	±	Á	Ñ	á	ñ
2	¢	²	Â	Ò	â	ò
3	£	³	Ã	Ó	ã	ó
4	¤	´	Ä	Ô	ä	ô
5	¥	µ	Å	Õ	å	õ
6	¦	¶	Æ	Ö	æ	ö
7	§	·	Ç	×	ç	÷
8	¨	¸	È	Ø	è	ø
9	©	¹	É	Ù	é	ù
A	ª	º	Ê	Ú	ê	ú
B	«	»	Ë	Û	ë	û
C	¬	¼	Ì	Ü	ì	ü
D		½	Í	Ý	í	ý
E	®	¾	Î	Þ	î	þ
F	¯	¿	Ï	ß	ï	ÿ

Table 11 G1 Code Set

20

Exhibit 15 Page 235

SONY0005752

7.4.5 G2 Code Set - Extended Miscellaneous Characters

The G2 code set contains the extended miscellaneous characters. See Table 12. Characters in the G2 space are transmitted by preceding them with the control character **EXT1** (10h) from the C0 character set. Following the **EXT1** extender prefix, the G2 characters are addressed with the base in the range 20h - 7Fh.

The unshaded characters in the G2 table below are those which exist in the Microsoft Windows character map. They are positioned in this code set to reflect their relative positioning in the MS Windows map. In Windows, these characters are mapped to the code set in the range 80h to 9Fh.

The **TSP** character (20h) represents a transparent space. This character has no text foreground or background color; i.e., it passes through the fill color of the window containing it.

The **NBTSP** character (21h) represents a non-breaking transparent space. This character is the same as a transparent space (TSP) with the exception that it should be used between words that should not be split when word-wrapping is in affect.

The ▌ character (30h) is a solid block which fills the entire character position with the text foreground color.

Also included in this set are the block drawing characters, opening and closing single and double quote marks, the trade mark symbol, the service mark symbol, and the remaining Latin-1 characters.

	G 2					
b7-b4 ⟍ b3-b0	2	3	4	5	6	7
0	TSP	▌				
1	NBTSP	'				
2		,				
3		"				
4		"				
5	...	•				
6						$^1/_8$
7						$^3/_8$
8						$^5/_8$
9		TM				$^7/_8$
A	Š	š				⎸
B						⌐
C	Œ	œ				⌊
D		SM				—
E						
F		Ÿ				⌈

Table 12 G2 Code Set

21

Exhibit 15 Page 236

SONY0005753

7.4.6 G3 Code Set - Future Expansion

The G3 code set is reserved for future expansion. It currently contains the single caption icon [CC] . See Table 13.

Characters in the G3 space are transmitted by preceding them with the control character **EXT1** (10h) from the C0 code set. Following the **EXT1** extender prefix, the G3 characters are addressed with the base codes in the range A0h - FFh.

b7-b4 \ b3-b0	A	B	C	D	E	F
0	[CC]					
1						
2						
3						
4						
5						
6						
7						
8						
9						
A						
B						
C						
D						
E						
F						

Table 13 G3 Code Set

22

Exhibit 15 Page 237

SONY0005754

7.4.7 C2 Code Set - Extended Control Code Set 1

The C2 code set is reserved for future extended miscellaneous control and caption command codes. See Table 14. Codes in the C2 space are transmitted by preceding them with the control character **EXT1** (10h) from the C0 code set. Following the **EXT1** extender prefix, the C2 characters are addressed with the base codes in the range 00h - 1Fh.

These codes can be succeeded by additional data bytes per the following:

      Codes 00h through 07h are single-byte control codes (0 - additional bytes).
      Codes 08h through 0Fh are two-byte control codes (1 - additional byte).
      Codes 10h through 17h are three-byte control codes (2 - additional bytes).
      Codes 18h through 1Fh are four-byte control codes (3 - additional bytes).

Example: The total sequence for a four-byte control code would be:
      EXT1, 18h, <data1>, <data2>, <data3>

DTVCC decoders which do not implement these commands must use their implied sizes to skip over them in the Service Blocks.

Table 14 C2 Code Set

Exhibit 15 Page 238

SONY0005755

7.4.8 C3 Code Set - Extended Control Code Set 2

The C3 code set is an additional set reserved for future extended miscellaneous control and caption command codes. See Table 15. Codes in the C3 space are transmitted by preceding them with the control character **EXT1** (10h) from the C0 character set. Following the **EXT1** extender prefix, the C3 command codes are addressed with the base codes in the range 80h - 9Fh.

Codes 80h through 8Fh are reserved for fixed-sized commands. These codes can be succeeded by additional data bytes per the following:

Codes 80h through 87h are five-byte control codes (4 - additional bytes).
Codes 88h through 8Fh are six-byte control codes (5 - additional bytes)

Example: The total sequence for a six-byte control code would be:
EXT1, 88h, <data1>, <data2>, <data3>,<data4>,<data5>

DTVCC decoders which do not implement these commands must use their implied sizes to skip over them in the Service Blocks.

Codes 90 through 9Fh are reserved for variable length caption commands. Variable-length caption commands have a 1-byte header following the command code. This header contains a 2-bit Type field (b7 - b6) and a 6-bit Length field (b5 - b0). The Type field allows a command to be broken up into several segments in order to be transmitted across multiple Service Blocks. The Type field has the following possible values: 00 - Beginning of Command (BOC), 01 - Continuation of Command (COC), 11 - End of Command (EOC), and All of Command (AOC). The Length field ranges from 0 - 63 and indicates the number of data bytes following the header. Data bytes can have any 8-bit value. Only one variable-length caption command per service can be transmitted at a time, and the command segments must be transmitted in order. This variable-length command capability is intended for commands which require the downloading of large units of data (e.g., fonts and graphics). DTVCC decoders which do not implement these commands must use the Length field to skip over them in the Service Blocks.

b7-b4	C	3
b3-b0	8	9
0		
1		
2		
3		
4		
5		
6		
7		
8		
9		
A		
B		
C		
D		
E		
F		

Table 15 C3 Code Set

24

8 DTVCC Interpretation Layer

The DTVCC Interpretation Layer defines the DTVCC Graphical User Interface. This discussion includes how the caption data coding is to be formatted when encoded and how it is to be interpreted when decoded. While the Caption Data Services Coding Layer (Section 7) identifies how service data bytes are represented, the Interpretation Layer describes how these bytes of data are to be processed. The data bytes for each caption service are interpreted as a unique data stream, independent from the other services.

This section defines all required and optional DTV closed-captioning features. Refer to Section 9 for a list of features required within a minimum DTVCC receiver implementation.

8.1 DTVCC Caption Components

The five major components of DTVCC captioning are Caption Screen, Caption Windows, Caption Pens, Caption Text, and Caption Display Synchronization.

- **Caption Screen**: The Caption Screen is the canvas on which caption windows are displayed.

- **Caption Windows**: The heart of caption definition consists of Caption Windows within which caption text is displayed.

- **Caption Pens**: The Caption Pens component defines styles and attributes for the appearance of the text within the caption windows.

- **Caption Text**: The Caption Text component defines how text is encoded and directed to specific windows.

- **Caption Synchronization**: The Caption Synchronization component controls the flow of interpretation of commands and caption text within the independent service data streams.

The following subsections provide a general overview of these components. These subsections are then followed by a detailed presentation of the DTVCC caption command set.

8.2 Screen Coordinates

A set of coordinates is defined to map a rectangular grid onto the "safe-title" area of the screen. This grid is used as a reference for superimposing captions. This reference is used to specify the position of caption windows.

Receivers which decode the DTV bit stream may have either a 16:9, 4:3 or other display screen aspect ratio. The coordinate-system grid size for a 16:9 receiver is 210 horizontal cells by 75 vertical cells. The 4:3 coordinate-system grid size is 160 horizontal cells by 75 vertical cells. For all other aspect ratio formats, a percentage or relative positioning coordinate system may be used such that a grid is mapped to the screen with 100 horizontal cells and 100 vertical cells.

The grid coordinates are specified as a pair of values in the form: (horizontal, vertical). The origin is the point in the upper left-most corner of the safe-title area, and is assigned the coordinate (0, 0). For the 16:9 format, the upper-right corner is (209, 0), the lower-left corner is (0, 74), and the lower right corner is (209, 74). A similar set of reference points is defined for the 4:3 format's 160 x 75 coordinate system: respectively, (0,0), (159, 0), (0, 74), and (159, 74). It is important to remember that these grid cells are not intended for text positioning, but for window positioning.

Once the window is positioned, the starting positioning of the text rows and columns follow, depending upon the size of the displayed font. The ending of the text rows depends upon the spacing (monospace vs. proportional space) of the chosen font.

Figure 11 shows an exaggerated relationship between a 16:9 screen, the 210 x 75 grid, the overscan area, the viewable display area, the safe-title area, and the caption windows.

25

Exhibit 15 Page 240
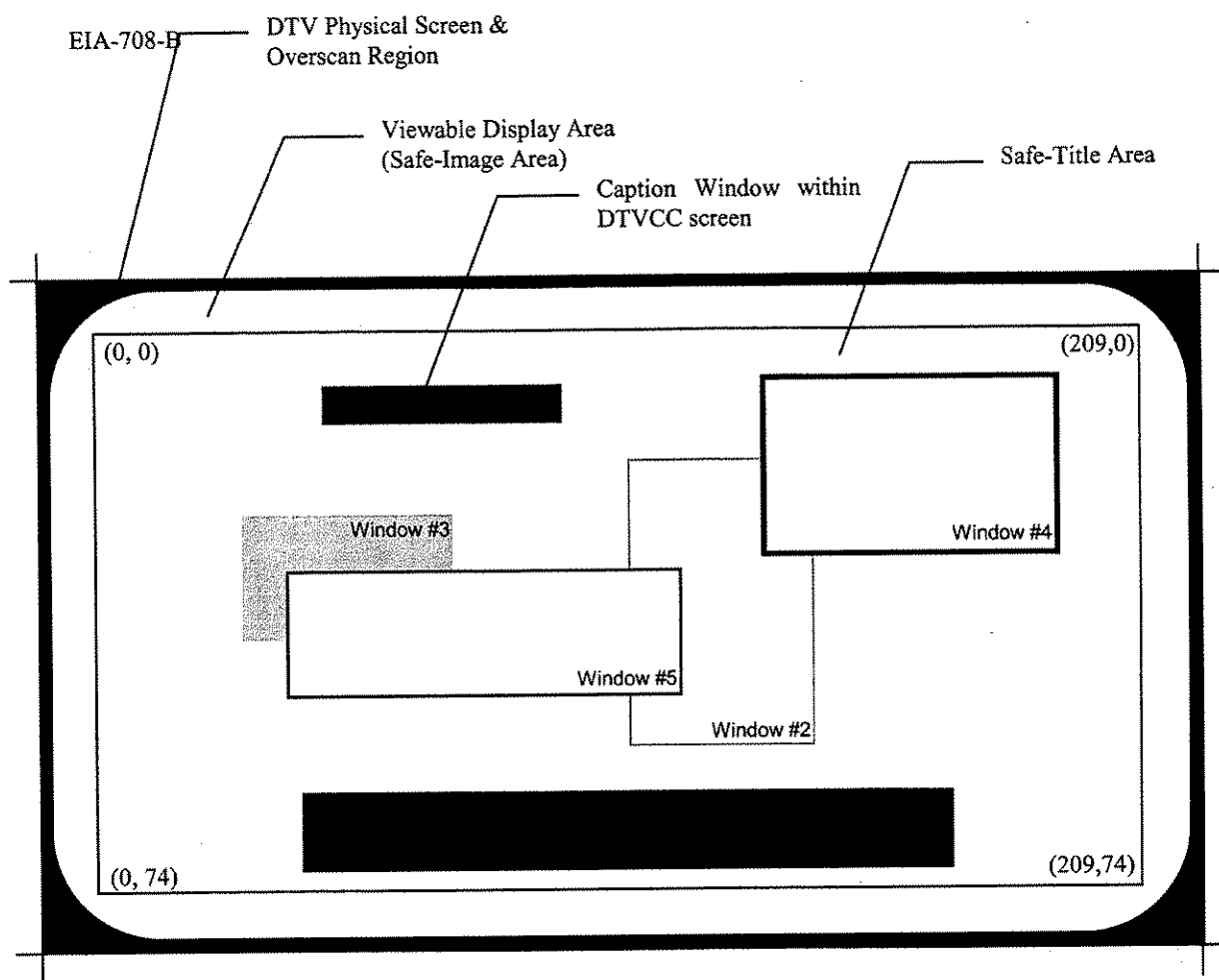
Figure 11 DTV 16:9 Screen and DTVCC Window Positioning Grid

8.3 User Options

Receiver manufacturers have the option to provide controls which may allow users to override styles and attributes specified in the service channel caption streams. Optional user controls might consist of caption font size, caption color and caption intensity (e.g., brightness) overrides. For further discussion, see the minimum DTVCC receiver decoder manufacturer recommendations in Section 9.

8.4 Caption Windows

All caption text is displayed and manipulated in receivers in the context of caption windows. There are 8 possible windows per service in which caption providers may write caption text. These windows may be implemented as buffers within receivers where any, none or all can be displayed at the same time. All 8 windows are available for the service currently selected by the user.

Manufacturers have the option of maintaining multiple sets of window buffers (i.e., instead of a single set of 8 buffers). Each set would be assigned to a service so that window processing of multiple services could occur simultaneously. This feature would have the effect that when a user switches services, the previously acquired and processed service data would be presented immediately. If only one set is used, the window buffers would have to be deallocated during service switching, and the new service would not appear until the buffers are reallocated and sufficient new service data are received.

The dimensions of a unique window specify an area on the screen which may contain caption text. The size of the window is based on the font size (SMALL, STANDARD, or LARGE) that the user has selected. Caption text designated for the window may not exceed the boundaries of the window, regardless of the font size the caption provider has specified or the font size the user has chosen.

26

Exhibit 15 Page 241

SONY0005758

A window's size may change on screen when a user changes the font size at the receiver. The effects of this window sizing are described further below.

8.4.1 Window Identifier
Each of the eight windows (and window buffers) is uniquely addressed by its window ID. Window ID numbers range from 0 to 7.

8.4.2 Window Priority
Each window has an associated priority which affects how it is displayed in conjunction with other windows that may be displayed at the same time. A higher priority (with 0 being the highest and 7 being the lowest) displayed window will overlap lower priority displayed windows on the screen.

8.4.3 Anchor Points
There are 9 locations within a window which serve as "anchors". An anchor specifies the reference point for positioning the window on the screen, and the "shrink and grow" directions (see Figure 12) of the window and caption text within the window when a user changes the font size.

8.4.4 Anchor ID
The 9 window anchor points are addressed by an Anchor ID which ranges from 0 to 8. Anchor ID 0 refers to the top-left corner of a window. Anchor 8 specifies the bottom-right corner of a window. Anchor 4 specifies the middle in the window.
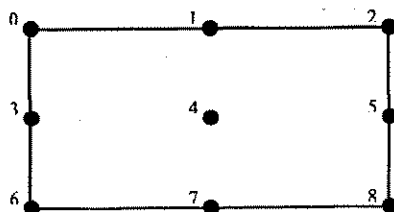
Figure 12 Anchor Points

Anchor points specify bounded and unbounded areas of caption text expansion and compression when the user overrides the standard font size for caption text display. Figure 13 shows the directions of expansion and compression of caption text for each anchor point. Solid lines indicate the bounded edges of the caption window. Dashed lines indicate the unbounded directions in which the caption text may shrink or grow.

Exhibit 15 Page 242

SONY0005759
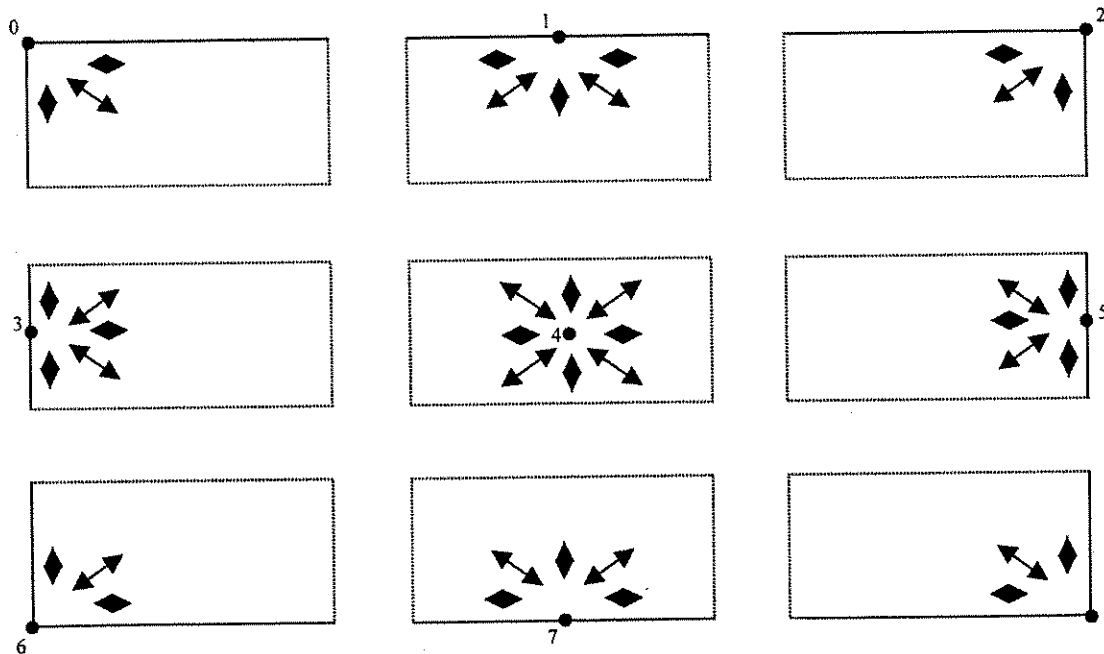
Figure 13 Implied Caption Text Expansion Based on Anchor Points

8.4.5 Anchor Location
The anchor location specifies where (in grid coordinates) the window's anchor point is to be physically located, and thus, the window itself. These grid coordinates are described in Section 8.2.

8.4.6 Window Size
The window size is specified in numbers of character rows and character columns for all display formats (16:9, 4:3, etc.). For all display formats, 15 is the maximum character row count and 32 is the maximum character column count.

> NOTE--A 16:9 format could handle longer rows (i.e., up to 42), but caption providers should keep the window column size at 32 characters, or less, in order to leave room on the display when the user selects the LARGE font size (see Section 8.4.7).

As for the physical size of the window on the screen, the receiver scales the window based on the "effective font size". The effective font size is based on a combination of the pen size chosen by the caption provider and the font size chosen by the receiver user. The height of the window is calculated as the number of rows multiplied by the physical height of the tallest character in the effective font size. The width of the window is calculated as the number of columns multiplied by the physical width of the widest character in the effective font size.

8.4.7 Window Row and Column Locking
The "lock rows" and "lock columns" window parameters fix the maximum number of rows and columns of caption text that a window may have. If a row or column parameter is "unlocked", the receiver may automatically add or adjust columns or rows to a window under certain circumstances.

This means that in a text-mode or roll-up type caption service (with no embedded carriage returns) where the user has specified a font size smaller than intended by the caption provider, more rows and columns could fit into the physical window (as it appears on the screen) than specified in the original window size parameters (see examples below).

28

Exhibit 15 Page 243

SONY0005760

When the user has specified a font size larger than intended by the caption provider, the window's width may grow larger if the columns are locked. In order to insure that there is enough room on the display for 16:9 formats to grow a caption row so that it will fit on the screen, caption providers should not create windows or caption rows with more than 32 character columns. Since 16:9 formats can display 42 characters per row and if the widest character in the large font is no more than 1.3 (i.e., 42/32) times larger than the widest character in the standard font, the enlarged caption (and window) should fit on the screen without word wrapping.

8.4.7.1 Effects When Choosing Smaller Font

Figure 14 provides examples of the effects row and column locking have on a window and its STANDARD-sized caption text when a receiver user chooses the SMALL font.

0

NOW IS THE WINTER OF OUR
DISCONTENT MADE GLORIOUS
SUMMER BY THIS SUN OF YORK.

Original Caption : Standard Font Size in 3 row x 28 col window

---

0

NOW IS THE WINTER OF OUR
DISCONTENT MADE GLORIOUS
SUMMER BY THIS SUN OF YORK;

1. Shrink with Rows and Columns locked

0

NOW IS THE WINTER OF OUR
DISCONTENT MADE GLORIOUS
SUMMER BY THIS SUN OF YORK;
AND ALL THE CLOUDS THAT

2. Shrink with Rows unlocked and Columns locked

0

NOW IS THE WINTER OF OUR DISCONTENT
MADE GLORIOUS SUMMER BY THIS SUN
OF YORK; AND ALL THE CLOUDS THAT
LOWERED UPON OUR HOUSE IN THE DEEP

3. Shrink with Rows and Columns unlocked

0

NOW IS THE WINTER OF OUR DISCONTENT
MADE GLORIOUS SUMMER BY THIS SUN
OF YORK; AND ALL THE CLOUDS THAT

4. Shrink with Rows locked and Columns unlocked

Figure 14 Examples of Caption Window Shrinking when User Picks Smaller Font

In the Figure 14 examples, the shaded rectangle indicates the size of the original window.

In example 1, the caption text and window shrink to a size in direct proportion to the original window. The caption contains the same number of lines and columns as the original, and the same words appear on the same line.

In example 2, the window indicates that there is room for another row of caption text. The window's height remains the same size.

In example 3, the window width and height remain the same; and with a smaller font, more characters may fit on a row. Thus, words are shifted up into the previous rows.

In example 4, the window shrinks in the vertical direction in order to maintain the same number of rows. The window height remains the same.

Exhibit 15 Page 244

SONY0005761

8.4.7.2 Effects When Choosing Larger Font

Figure 15 provides examples of the effects row and column locking have on a window and its STANDARD-sized caption text when a receiver user chooses the LARGE font.

NOW IS THE WINTER OF OUR
DISCONTENT MADE GLORIOUS
SUMMER BY THIS SUN OF YORK.

Original Caption : Standard Font Size in 3 row x 28 col window

NOW IS THE WINTER OF OUR
DISCONTENT MADE GLORIOUS
SUMMER BY THIS SUN OF YORK.

1. Grow with Rows and Columns locked

NOW IS THE WINTER OF
OUR DISCONTENT MADE
GLORIOUS SUMMER BY
THIS SUN OF YORK.

2. Grow with Rows unlocked and Columns locked

NOW IS THE WINTER OF OUR
DISCONTENT MADE GLORIOUS
SUMMER BY THIS SUN OF YORK.

3. Grow with Rows and Columns unlocked

NOW IS THE WINTER OF OUR DISCONTENT MADE
GLORIOUS SUMMER BY THIS SUN OF YORK.

4. Grow with Rows locked and Columns unlocked

Figure 15 Examples of Caption Window Growing when Going to Larger Font

In the Figure 15 examples above, the shaded rectangle indicates the size of the original window.

In example 1, the caption text and window grow to a size in direct proportion to the original window. The caption contains the same number of lines and columns as the original, and the same words appear on the same line.

In example 2, the window width remains the same, but enough rows are automatically added to accommodate all of the caption text.

In example 3, the window is unrestricted in growth either in width or height. The decoder manufacturer is free to format the caption in anyway desired. However, since this allows for much ambiguity, it is suggested that the caption be formatted as in Example 1.

In example 4, the window height remains the same, but enough columns are automatically added to accommodate all of the caption text.

8.4.8 Word Wrapping

Whenever possible, the receiver should render embedded carriage returns as line breaks, since these carriage returns indicate an important aspect of the caption's formatting as determined by the service provider. However, it may sometimes be necessary for the receiver to ignore embedded line breaks. For example, if a caption is to appear in a larger font, and if its window's rows and/or columns are unlocked, the rows of text may need to become longer or shorter to fit within the allocated space. Such automatic reformatting of a caption is known as "word wrap."

30

Exhibit 15 Page 245

SONY0005762

The receiver should follow standard typographic practice when implementing word wrap. Potential breaking points (word-wrapping points) are indicated by the space character (20h) and by the hyphen character (2Dh).

If a row is to be broken at a space, the receiver should remove the space from the caption display. If a row is to be broken after a hyphen, the hyphen should be retained.

If an embedded return is to be removed, it should usually be replaced with a space. However, if the character to the left of the embedded return is a hyphen, the embedded return should be removed but NOT replaced with a space.

This specification does not include optional hyphens, nor does it provide for any form of automatic hyphenation. No non-breaking hyphen is defined. The non-breaking space (A0h in the G1 code set) and the non-breaking transparent space (21h in the G2 code set) should not be considered as potential line breaks.

If a single word exceeds the length of a row, the word should be placed at the start of a new row, broken at the character following the last character that fits on the row, and continued with further breaks if needed.

8.4.9 Window Text Painting
This section defines window text painting parameters and the interactions between them.

8.4.9.1 Justification
Caption text shall be formatted within the window based upon the justification type. The nomenclature for the justification types is based on standard English left-to-right Print Direction (see Section 8.4.9.2), and should be interpreted as follows:

Left Justification -- Text is justified to the start of the row, e.g., the left side for left-to-right print direction, the bottom for bottom-to-top print direction, and so on.

Right Justification -- The opposite of left justification. The text is aligned to the end of the row.

Centered Justification -- The text is centered horizontally when the print direction is left-to-right or right-to-left, and centered vertically when the print direction is top-to-bottom or bottom-to-top.

Full Justification -- The text is aligned to the left and right margins when the print direction is left-to-right or right-to-left, and to the top and bottom margins when the print direction is top-to-bottom or bottom-to-top.

8.4.9.2 Print Direction
The Print Direction parameter specifies in which direction characters will be written on a (horizontal or vertical) caption row (left-to-right, right-to-left, top-to-bottom, or bottom-to-top).

Table 16 defines how the cursor should move after drawing a character (when Justification is "Left" -- see Section 8.4.9.1) or after receiving a carriage return for each combination of Print Direction and Scroll Direction (see Section 8.4.9.3). Combinations of Print Direction and Scroll Direction that are not listed in this table are not permitted.

Exhibit 15 Page 246

SONY0005763

Print Direction	Scroll Direction	Cursor Movement	Carriage Return Behavior
Left -> Right	Top-> Bottom	increment column	decrement row, column=0
Left -> Right	Bottom->Top	increment column	increment row, column=0
Right->Left	Top-> Bottom	decrement column	decrement row, column=max
Right->Left	Bottom->Top	decrement column	increment row, column=max
Top-> Bottom	Left -> Right	increment row	decrement column, row=0
Top-> Bottom	Right->Left	increment row	increment column, row=0
Bottom->Top	Left -> Right	decrement row	decrement column, row=max
Bottom->Top	Right->Left	decrement row	increment column, row=max

Table 16 Cursor Movement After Drawing Characters

Figure 16 shows how a 3-row caption would appear using various justifications (see Section 8.4.9.1), Print Directions and Scroll Directions (see Section 8.4.9.3).


```
Left justified          Left justified          Right justified         Right justified
Left->Right print       Right->Left print       Left->Right print       Right->Left print
Bottom ->Top scroll     Bottom ->Top scroll     Top ->Bottom scroll     Top ->Bottom scroll
ROW  ONE                       ENO  WOR               AND  THREE         EERHT  DNA
TWO                                 OWT                      TWO         OWT
AND  THREE                    EERHT  DNA                ROW ONE         ENO  WOR
```

```
Left justified          Right justified         Center justified        Center justified
Top->Bottom print       Bottom ->Top print      Left->Right print       Top->Bottom print
Right->Left scroll      Left->Right scroll       Bottom ->Top scroll     Left->Right scroll
RTA                     EOE                        ROW  ONE              A
OWN                     EWN                          TWO                 N  R
WOD                     RTO                        AND  THREE            D  O
                        H                                                   TW
O   T                   T  W                                             TW
N   H                      O                                             HOO
E   R                   D  R                                             R  N
    E                   N                                               E  E
    E                   A                                               E
```

Figure 16 Examples of Various Justifications, Print Directions and Scroll Directions

8.4.9.3 Scroll Direction

The Scroll Direction parameter specifies in which direction the text will scroll (left-to-right, right-to-left, top-to-bottom, or bottom-to-top) when carriage returns and word wrapping is encountered. For example, NTSC style roll-up captions are achieved by specifying left-to-right printing with bottom-to-top scrolling.

Horizontal scrolling is allowed only with vertical print directions, and vertical scrolling with horizontal print directions (see Section 8.4.9.2). For example, left-to-right scrolling may be used with top-to-bottom or bottom-to-top printing, but not with left-to-right or right-to-left printing.

8.4.9.4 Combining Text Painting Attributes

This section describes one of a multitude of text painting effects using the various window attributes.

A 2-line ticker-tape effect can be accomplished with the following parameter settings: print direction set to "top-to-bottom" with scroll direction set to "right-to-left", and window size set to 2 rows and X columns. The initial pen location is set to the upper right corner of the window. One character from row 1 is sent, followed by 1 character from row 2, and a carriage return, and so on. The first character for row 1 will appear on the screen by itself. With top-to-bottom print direction, the 1st character for row 2 will appear beneath the first row 1 character. The carriage return is sent, creating a new column and causing the existing 2 characters to scroll to the left and the next row 1 character to appear to the right of the existing row 1 character.

Exhibit 15 Page 247

SONY0005764

8.4.10 Window Display

Caption text can be written to a window whether it is currently displayed (i.e., visible) or not. Writing a whole caption to a non-displayed window and then sending a **DisplayWindows** command will cause the caption to "pop-on" (if the SNAP display effect has been set for the window). To achieve successive "pop-on" type captions, two windows are required (just as displayed and non-displayed memory buffers are used in NTSC captioning), with the required sequencing of **DisplayWindows** and **HideWindows** commands to maintain the effect.

When a window is displayed or hidden, a Display Effect parameter can be specified which controls whether the window snaps on/off ("pop-on"), fades on/off, or wipes on/off. The rate of fade on/off may be a manufacturer's option, but an effect speed can be specified with the **SetWindowAttributes** command. The direction (left-to-right wipe, right-to-left wipe, top-to-bottom wipe, and bottom-to-top wipe) in which a window wipes on/off can also be specified with this command.

8.4.11 Window Colors and Borders

The area within a window can have different characteristics regarding color and opacity. Windows can be clear, or filled with a specified color. A window can be filled with an opaque color from a color palette (see Section 8.8). The colors within a window can have various effects associated with them. The window can be transparent (i.e., clear - that is, letting the underlying video show through), translucent (i.e., the underlying video is passed through a fixed level of color background filtering so that underlying video may partially show through the window), solid (opaque with a nominal level of saturation), or flashing (alternating transparency and opacity at the nominal, or solid, level of saturation).

Windows may have no borders, or be bounded by an enclosing border. Borders may be raised, depressed, uniform, or drop-shadowed (left or right). Manufacturers have a bit of latitude in how these border effects are achieved. A uniform border may appear as a single line surrounding the window. Raised, depressed and shadowed borders can be used to achieve 3-dimensional effects.

8.4.12 Predefined Window and Pen Styles

Colors, text painting, effects, and border types can be customized with the **SetWindowAttributes** and **SetPenAttributes** commands. However, the caption provider may wish to use predefined standard windows styles. A set of predefined styles will be hard stored in receivers. This set will anticipate the most widely used types of caption windows in order to conserve caption channel bandwidth by eliminating the need to transmit superfluous **SetWindowAttributes** and **SetPenAttributes** commands.

Predefined window and pen styles can be specified by the *window style* and *pen style* ID parameters in the **DefineWindow** command. See Section 9.12 for the defined "predefined window and pen styles".

8.5 Caption Pen

The caption text within a window is written with a *pen*. A pen governs the size, font, colors, and styles of the text within a window. Pen attributes are specified separately from window attributes through the use of the **SetPenAttributes** command. A window may contain text with more than one combination of pen attributes (i.e., different fonts, colors, etc.). Pen characteristics remain constant for a window unless specifically changed (via a subsequent **SetPenAttributes** command) by the caption provider. Pen attribute changes only affect text written after the change (i.e., the **SetPenAttributes** command does not change existing text within a window) and until a new **SetPenAttributes** command is encountered.

8.5.1 Pen Size

TV receivers may implement different sized characters (e.g., small, standard, and large) for supported fonts. Caption providers specify the pen size to be displayed, but users may override the size of the characters (i.e., which font size is to be viewed on the receiver) as desired.

8.5.2 Pen Spacing

Monospacing and proportional spacing are inherent attributes of the font chosen to write caption text. They cannot otherwise be controlled by either the caption provider or the user.

8.5.3 Font Styles

Caption providers may specify 1 of 8 different font styles to be used to write caption text. The styles specified in the "font style" parameter of the **SetPenAttributes** command are numbered from 0 through 7.

33

Exhibit 15 Page 248

SONY0005765

The following is a list of the 8 recommended font styles. For information purposes only, each font style references one or more popular fonts which embody the characteristics of the style:

> 0 - Default (undefined)
> 1 - Monospaced with serifs (similar to Courier)
> 2 - Proportionally spaced with serifs (similar to Times New Roman)
> 3 - Monospaced without serifs (similar to Helvetica Monospaced)
> 4 - Proportionally spaced without serifs (similar to Arial and Swiss)
> 5 - Casual font type (similar to Dom and Impress)
> 6 - Cursive font type (similar to Coronet and Marigold)
> 7 - Small capitals (similar to Engravers Gothic)

Implementation of these fonts styles by decoder manufacturers is optional. Font styles which are not supported in a decoder should be displayed in an available font which is most similar to the requested font style in the **SetPenAttributes** command.

8.5.4 Character Offsetting
Characters can be positioned relative to the row baseline in one of three ways: subscript (offset vertically downward), superscript (offset vertically upward), or normal (no offset).

8.5.5 Pen Styles
Pen styles can be italicized and/or underlined.

8.5.6 Foreground Color and Opacity
The foreground color and opacity of the caption characters can be specified by the caption provider. The character foreground opacity can be set to transparent (i.e., showing underlying video), translucent (i.e., showing a filtered level of underlying video), solid, or flashing.

8.5.7 Background Color and Opacity
Characters are individually contained within a small, rectangular background box. These background boxes have color and opacity attributes which may be specified separately from character foreground attributes.

8.5.8 Character Edges
The color attributes of the edges (or outlines) of the character foregrounds may be specified separately from the character foreground and background. Edge opacities have the same attribute as the character foreground opacities. The type of edge surrounding the body of the character may be NONE, RAISED, DEPRESSED, UNIFORM, or DROP_SHADOWED.

8.5.9 Caption Text Function Tags
Caption text can be tagged with a marker indicating the type of text content that is being encoded. This tagging is performed by caption providers via the **SetPenAttributes** command. With this command, caption text can be tagged with any one of the following qualities:

> 0 - **Dialog** (normal words being spoken by characters in the programming)
> 1 - **Source or speaker ID** (name of the speaker, or a description of the source of a sound)
> 2 - **Electronically reproduced voice** (spoken audio heard by the characters in the drama coming from a phone, radio, PA, etc.)
> 3 - **Dialog in a language other than the drama's primary language**
> 4 - **Voiceover** (narration or other disembodied voice NOT heard by the characters in the drama)
> 5 - **Audible Translation** (voice of a disembodied translator NOT heard by the characters in the drama)
> 6 - **Subtitle Translation** (text showing a translation into the primary language of the drama)
> 7 - **Voice quality description** (description of a voice quality)
> 8 - **Song Lyrics** (words being sung)
> 9 - **Sound effect description** (a description of a nonverbal sound or music heard by the characters in the drama)

Exhibit 15 Page 249

SONY0005766

10 - Musical score description (a description of background music NOT heard by the characters in the drama)
11 - Expletive (an interjectory word or expression, possible profane or harsh)
12 to 14 - (undefined)
15 - Text not to be displayed (reserved for future use by a text-based control and information channel within the caption text stream; e.g., hypertext, related non-caption program information)

In the previous list, all of the tagged text is to be displayed in the current caption window as indicated, except for tag 15 ("Text not to be displayed"), and optionally, tag 11 ("Expletive").

Decoder manufacturers have the flexibilty to display tagged text in an number of ways desired in order to enhance the caption viewing experience. For example, "Source or speaker ID" text may always be automatically displayed in italics by the decoder. This may be a feature which the viewer can enable and disable.

If no other information is available, decoders are to default caption text to the "Dialog" tag (0). That is, if caption text is received for a window when no **SetPenAttributes** command has been received, the text is to be treated as "Dialog".

8.6 Caption Text

Caption text is always written to the current caption window with the attributes set with the preceding **SetPenColor** and **SetPenAttributes** commands. There is no specific Text Write caption command; any displayable characters or codes from the G0, G1, G2, or G3 code sets in Service Blocks which are not part of any caption command are considered text to be written to the current window.

Caption text sequences must be terminated by either the start of a new DTVCC Command, or with an ASCII **ETX** (0x03) character when no other DTVCC Commands follow. This requirement aids decoders in processing text sequences when text spans multiple service blocks.

8.7 Caption Positioning

The starting row and column position of an individual character or set of characters may be specified at any time via the **SetPenLocation** command. The position of subsequent characters written after a location is specified depends upon the Print Direction and Scroll Direction specified for a window.

Character positions specify memory locations within the internal buffer holding the window text, and thus, do not specify physical locations on the screen. Screen locations are dependent on a multitude of pen and window attributes (e.g., character size, character spacing, justification, and print direction), and whether or not the font is proportionally spaced.

8.8 Color Representation

Foreground and background colors are specified in the Caption Commands as combinations of the red-green-blue color triad. Two bits are specified for each red, green, and blue color value which defines the intensity of each individual color component. Colors are specified as a group; i.e., (<red>, <green>, <blue>). The range of color specification is (0, 0, 0) [for black] through (3, 3, 3) [for bright white]. Bright red has a color value of (3, 0, 0); bright green has a color value of (0, 3, 0); and bright blue has a color value of (0, 0, 3). This coding scheme provides 64 different colors.

8.9 Service Synchronization

For the most part, caption providers insert caption commands and text into the DTVCC Caption Channel stream in a real-time manner. That is, captions are transmitted shortly before they are to be displayed. This is the technique used in NTSC captioning.

DTVCC provides for an additional synchronization capability by allowing DTVCC data to be pre-sent and processed at a later time, all under the control of the decoder. The **Delay** command provides this added functionality.

8.9.1 Delay Command

Decoders must maintain a Service Input Buffer for each of the services which can be processed simultaneously. This input buffer has a minimum size of 128 bytes. All DTVCC data for a service pass through the Service Input Buffer.

Exhibit 15 Page 250

SONY0005767

Most of the time, the data falls through the buffer instantaneously, and the data are processed by the DTVCC decoder as they are received.

The **Delay** command is used to instruct the decoder to suspend processing of the Service Input Buffer data for a specified time period. This command has a time-out period specified as its parameter. This period is specified in tenths of seconds.

When a **Delay** command is encountered, the decoder waits for the specified delay time to expire before processing any other service data. During the delay interval, incoming data for the active service is buffered in the Service Input Buffer. When the delay interval expires, interpretation of the incoming data is resumed.

The **Delay** command may be used in instances where the caption provider knows that it is about to lose access to the DTVCC encoder-to-decoder stream. Prior to losing access, the caption provider pre-sends a set of DTVCC commands with one or more embedded **Delay** commands. When the caption provider loses access, the decoders process the sequencing of the buffered data according to the providers' instructions.

Any active delay interval is automatically canceled when the Service Input Buffer becomes full. The decoder begins interpreting buffered data in order not to lose any incoming data.

8.9.2 DelayCancel Command

Caption providers may override an active **Delay** command via the **DelayCancel** command. The **DelayCancel** command terminates any currently pending Delay command processing, and causes the decoder to begin processing any data in the Service Input Buffer.

The **DelayCancel** command must be detected (recognized) prior to the input of Service Input Buffer. That is, the **DelayCommand** is not buffered. If it were, it would not be processed until the delay interval expires.

The **Delay** and **DelayCancel** commands are analogous to Service Input Buffer processing "suspend" and "resume". These two commands allow caption providers to preload a set of commands and with a **Delay** command (suspend) to delay their output (using the maximum delay interval), and then issue a **DelayCancel** command (resume) at a desired instance to have the commands executed en mass.

8.9.3 Reset Command

The **Reset** command reinitializes a DTVCC service. The effects of re-initialization are described in Section 8.9.5. The **Reset** command is encoded by a caption provider to reset caption service processing within decoders.

It is recommended that a **Reset** command be issued at the beginning of a captioned program, or program segment. This ensures that decoders are initialized and ready for a new program, and erases any left-over windows and captions that were not deleted as a result of such events as video up-cutting during transitions from one program source to another.

8.9.4 Reset and DelayCancel Command Recognition

In order for the **DelayCancel** and service **Reset** commands to be effective, they must be recognized after they are retrieved from the Caption Channel Data Stream and prior to their routing to (insertion into) the Service Input Buffer. All other commands are interpreted when they are pulled from the Service Input Buffer.

Figure 17 shows this point of "pre-processing" for a decoder which simultaneously processes multiple service streams.

Exhibit 15 Page 251

SONY0005768
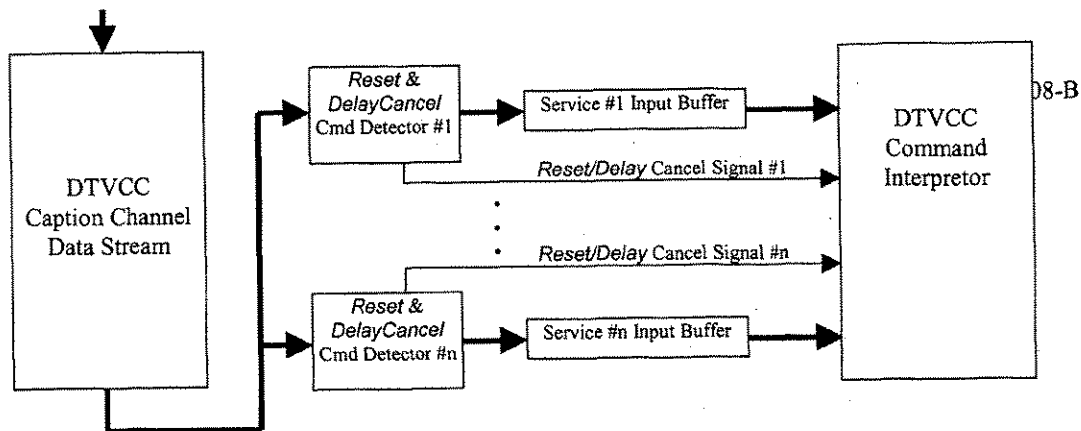
Figure 17 Reset & DelayCancel Command Detector(s) and Service Input Buffers

For descriptive purposes, Figure 18 shows a "hardware" implementation for the "*Reset & DelayCancel* Command Detector(s)"; a software implementation should be straight-forward. The Detectors may be implemented as a pair of 8-bit comparators. Since each of these commands is only a single byte in length (**Reset = 0x8F, DelayCancel = 0x8E**), this comparison may be simple, as is shown in Figure 18.
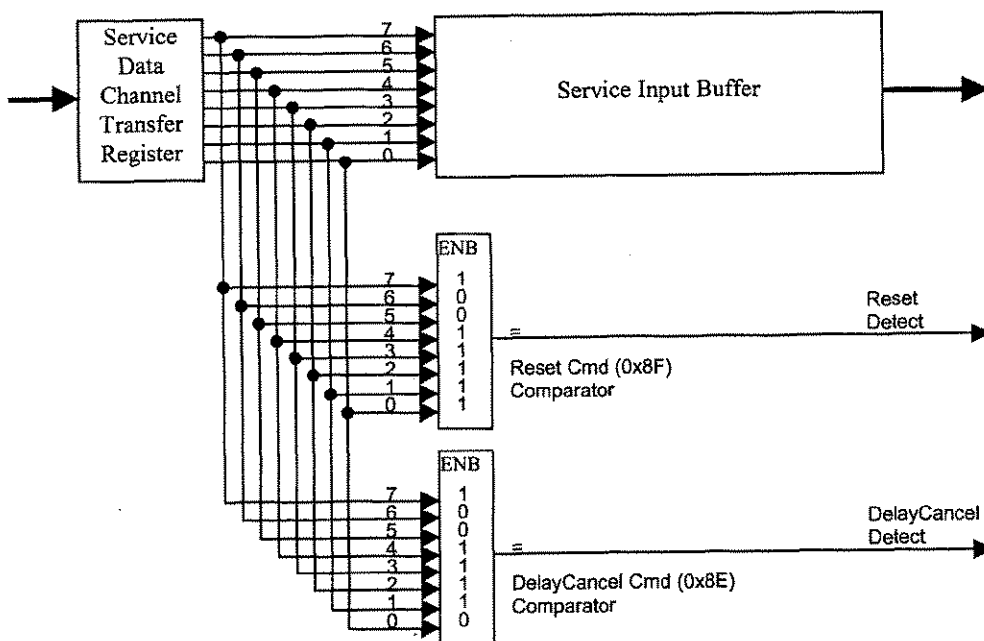
Figure 18 Reset & DelayCancel Command Detector(s) Detail

8.9.5 Service Reset Conditions

The "resetting" or "re-initialization" of a service means:

- that the service's windows are removed from the display,
- all defined windows for the service are deleted,
- all window and pen attributes for the service are deleted, and
- the Service Input Buffer is cleared.

37

Exhibit 15 Page 252

SONY0005769

A service shall be reset when any one of the following events occur:

1. a **Reset** command is received for a service.
2. a channel change occurs.
3. the Service Input Buffer overflows.
4. A loss of continuity in the Caption Channel Packet sequence_number.

8.10 DTVCC Command Set

This section presents the commands which may be encoded by caption providers. The commands are grouped into the following command types: Window Commands, Pen Commands, Caption Text Commands, and Synchronization Commands.

8.10.1 Window Commands

These commands create, delete, modify, and display windows, and specify the current caption window for a caption service.

Command Code	Command Name	Parameters
CW0, ... , CW7	SetCurrentWindow	window ID
DF0, ... , DF7	DefineWindow	window ID, priority, anchor point, relative positioning, anchor vertical, anchor horizontal, row count, column count, row lock, column lock, visible, window style ID, pen style ID
DLW	DeleteWindows	window map
DSW	DisplayWindows	window map
HDW	HideWindows	window map
TGW	ToggleWindows	window map
SWA	SetWindowAttributes	justify, print direction, scroll direction, wordwrap, display effect, effect direction, effect speed, fill color, fill opacity, border type, border color

8.10.2 Pen Commands

These commands define pen attributes and colors.

Command Code	Command Name	Parameters
SPA	SetPenAttributes	pen size, font, text tag, offset, italics, underline, edge type
SPC	SetPenColor	fg color, fg opacity, bg color, bg opacity, edge color
SPL	SetPenLocation	row, <column

8.10.3 Synchronization Commands

These commands control the rate of service data interpretation.

38

Exhibit 15 Page 253

SONY0005770

Command Code	Command Name	Parameters
DLY	Delay	tenths of seconds
DLC	DelayCancel	
RST	Reset	

8.10.4 Caption Text

There is no specific Text Write caption command. That is, any characters or codes from the G0, G1, G2, or G3 code sets in Service Blocks which are not part of any caption command are considered text to be written to the current window. Any such encountered text is written to the current window starting at the window's current cursor position. The window's current cursor is adjusted automatically to the row and column following the last character in the text string.

In order to assist decoders in determining the end of caption text segments for dangling segments at the end of a series of caption commands and text, an ETX code (0x03) from the C0 Code Space is to be inserted at the end of the text segment to terminate the segment. Text which is immediately followed by a caption command code (from the C1 Code Space) does not require the ETX code. That is, the decoder determines the end of a text segment when it encounters a caption command code, or an ETX code.

Without the ETX assist, decoding software may find it difficult to know if it should wait until the next service block to see if there is more caption text for a caption row when a service block terminates in text. If the last used byte in the block is an ETX, then the decoder knows there is no more text for this segment in a subsequent service block, and the decoder can process it as a single entity (this is especially helpful when doing Center justification of a row and word wrap processing, for example). If the last used byte in the block is a text character, then the decoder must wait until the next service block to see if there is any more text.

8.10.5 Command Descriptions

Each command is described in detail within this section.

Exhibit 15 Page 254

SONY0005771

SET CURRENT WINDOW - (CWx)

Name: SetCurrentWindow - Specify current window ID

Command Type: Window

Format: SetCurrentWindow (*window ID*)

Parameters: ■ *window ID* (id) is the unique window identifier (0 - 7).

Command Coding: CW0, ... , CW7 = 80h, ... , 87 (10000000b, ... , 10000111b)

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	0	0	id_2	id_1	id_0	command

Description: SetCurrentWindow specifies the window to which all subsequent window style and pen/text commands are directed. The *window ID* must address a window which has already been created by the **DefineWindow** command. This command initializes a "current window ID" variable internal to the decoder.
SetCurrentWindow directs the following commands to the specified window:
SetWindowAttributes, SetPenAttributes, SetPenColor, SetPenLocation, WriteText.

40

Exhibit 15 Page 255

SONY0005772

DEFINE WINDOW - (DF0 ... DF7)

Name: **DefineWindow** - Create window and set initial parameters

Command Type: Window

Format: **DefineWindow** (*window ID, priority, anchor point, relative positioning, anchor vertical, anchor horizontal, row count, column count, row lock, column lock, visible, window style ID, pen style ID*)

Parameters:

- *window ID* (id) is the unique window identifier (0 - 7)
- *priority* (p) is the window display priority (0 - 7)
- *anchor point* (ap) is the window anchor position number to use for the window's position on the screen (0 - 8).
- *relative positioning* (rp) is a flag that, when set to 1, indicates that the *anchor vertical* (av) and *anchor horizontal* (ah) coordinates specify "relative coordinates" (i.e., percentages) instead of physical screen coordinates.
- *anchor vertical* (av) is the vertical position of the window's anchor point on the viewing screen when the window is displayed (0 – 74 for 16:9 and 4:3 systems and the *relative positioning* (rp) parameter is set to 0, or 0-99 when the 'rp' parameter is set to 1).
- *anchor horizontal* (ah) is the horizontal position of the window's anchor point on the viewing screen when the window is displayed (0 - 209 for 16:9 systems, 0 - 159 for 4:3 systems and the *relative positioning* (rp) parameter is set to 0, or 0-99 when the 'rp' parameter is set to 1).
- *row count* (rc) is the number of virtual rows of text (assuming the STANDARD *pen size;* see **SetPenAttributes**) the window body will hold (0 - 11).
- *column count* (cc) is the number of virtual columns of text (assuming the STANDARD *pen size;* see **SetPenAttributes**) the window body will hold (0 - 31 for 4x3 formats, and 0 - 41 for 16x9 formats).
- *row lock* (rl), when set to YES, fixes the absolute number of rows of caption text the window will contain. When NO, *row lock* permits a receiver to add more rows to a window if the user selects a smaller size font other than intended by the caption provider. [YES, NO] = [1, 0].
- *column lock* (cl), when set to YES, fixes the absolute number of columns of caption text the window will contain. When NO, *column lock* permits a receiver to add more columns to a window if the user selects a smaller size font other than intended by the caption provider. [YES, NO] = [1, 0].
- *visible* (v), when set to YES, causes the window to be viewed (i.e., displayed) on the screen immediately after it is created. When set to NO, the window is not displayed (i.e., hidden) after it is created. [YES, NO] = [1, 0].
- *window style ID* (ws), when non-zero, specifies 1 of 7 static preset window attribute styles to use for the window when it is created (0 - 7). When zero during a window create, the window style is automatically set to window style #1. When zero during a window update, no window attribute parameters are changed. See **SetWindowAttributes** command.
- *pen style ID* (ps), when non-zero, specifies 1 of 7 static preset pen attribute styles to use for the window when it is created (0 - 7). When zero during a window create, the pen style is automatically set to pen style #1. When zero during a window update, no pen attribute parameters are changed. See **SetPenAttributes** command.

41

Exhibit 15 Page 256

EIA-708-B

Command Coding: DF0, ... , DF7 = 98h, ... , 9Fh (10011000b, ... , 10011111b)

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	1	1	id_2	id_1	id_0	command
0	0	v	rl	cl	p_2	p_1	p_0	parm1
rp	av_7	av_6	av_5	av_4	av_3	av_1	av_0	parm2
ah_7	ah_6	ah_5	ah_4	ah_3	ah_2	ah_1	ah_0	parm3
ap_3	ap_2	ap_1	ap_0	rc_3	rc_2	rc_1	rc_0	parm4
0	0	cc_5	cc_4	cc_3	cc_2	cc_1	cc_0	parm5
0	0	ws_2	ws_1	ws_0	ps_2	ps_1	ps_0	parm6

Description:

DefineWindow creates a window for the specified window identifier (*window ID*) and initializes the window with the non-style parameters listed in the command and with the available static style presets specified with the *window style ID* and *pen style ID* parameters. If the window is not already defined when the **DefineWindow** command is received, the window is created; otherwise it is simply updated. If the window is being created, all character positions in the window are set to the window fill color and the pen location is set to (0, 0). The **DefineWindow** command also makes the defined window the current window (see **SetCurrentWindow**).

When a window is created, a specific or automatic *window style ID* is assigned in order to preload a set of known window attribute values. These attributes can be subsequently modified with the **SetWindowAttributes** command. A *pen style ID* is assigned in the same fashion. Pen style attributes can be subsequently modified with the **SetPenAttributes** command.

When a decoder receives a **DefineWindow** command for an existing window, the command is to be ignored if the command parameters are unchanged from the previous window definition. Encoders or caption providers may periodically repeat window definitions in order for receivers to acquire a service and begin decoding and displaying captions with little delay. It is suggested that all window and pen definition and attribute commands be repeated in this way.

When an existing window is being updated (e.g., resized or moved) with the **DefineWindow** command, the pen location and pen attributes are unaffected.

42

Exhibit 15 Page 257

SONY0005774

CLEAR WINDOWS - (CLW)

Name:	**ClearWindows** - Clears Text from a set of windows
Command Type:	Window
Format:	**ClearWindows** (*window map*)
Parameters:	■ *window map* (w) is an 8-bit bitmap which specifies the window(s) affected by the command. Each bit position represents a window (i.e., *window ID*) to be affected (e.g., bit position 4 addresses the window with the *window ID* of 4). A value of 1 in a bit position specifies that the associated window is to be processed by the command. A bit value of 0 indicates that the associated window is unaffected by the command.

Command Coding: **CLW = 88h (10001000b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	0	1	0	0	0	command
w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	parm1

Description: **ClearWindows** removes any existing text from the specified window(s) in the window map. When a window is cleared, the entire window is filled with the window fill color.

43

Exhibit 15 Page 258

SONY0005775

DELETE WINDOWS - (DLW)

Name: DeleteWindows - Deletes window definitions for a set of windows

Command Type: Window

Format: **DeleteWindows** (*window map*)

Parameters: ■ *window map* (w) is an 8-bit bitmap which specifies the window(s) affected by the command. Each bit position represents a window (i.e., *window ID*) to be affected (e.g., bit position 4 addresses the window with the *window ID* of 4). A value of 1 in a bit position specifies that the associated window is to be processed by the command. A bit value of 0 indicates that the associated window is unaffected by the command.

Command Coding: **DLW = 8Ch (10001100b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	0	1	1	0	0	command
w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	parm1

Description: **DeleteWindows** removes all specified windows from the receiver. For example, a *window map* value of 64 (hex) deletes windows 6 (w_6), 5 (w_5), and 2 (w_2). A *window map* value of FF (hex) deletes all defined windows in the receiver. If the current window is deleted, then the decoder's current window ID is unknown and must be reinitialized with either the **SetCurrentWindow** or **DefineWindow** command.

Exhibit 15 Page 259

SONY0005776

DISPLAY WINDOWS - (DSW)

Name: **DisplayWindows** – Causes a set of windows to become visible

Command Type: Window

Format: **DisplayWindows** (*window map*)

Parameters: ■ *window map* (w) is an 8-bit bitmap which specifies the window(s) affected by the command. Each bit position represents a window (i.e., *window ID*) to be affected (e.g., bit position 4 addresses the window with the *window ID* of 4). A value of 1 in a bit position specifies that the associated window is to be processed by the command. A value of 0 indicates that the associated window is unaffected by the command..

Command Coding: **DSW = 89h (10001001b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	0	1	0	0	1	command
w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	parm1

Description: **DisplayWindows** causes the specified, existing windows to be visible on the receiver display screen. For example, a *window map* value of 96 (hex) displays windows 7 (w_7), 4 (w_4), 2 (w_2), and 1 (w_1). A *window map* value of FF (hex) causes all existing windows in the receiver to be displayed. This command does not affect the current window ID.

Exhibit 15 Page 260

SONY0005777

HIDE WINDOWS - (HDW)

Name: HideWindows – Causes a set of windows to become invisible

Command Type: Window

Format: **HideWindows** (*window map*)

Parameters:
■ *window map* is an 8-bit bitmap which specifies the window(s) affected by the command. Each bit position represents the window (i.e., *window ID*) to be affected (e.g., bit position 4 addresses the window with the *window ID* of 4). A value of 1 in a bit position specifies that the associated window is to be processed by the command. A value of 0 indicates that the associated window is unaffected by the command.

Command Coding: **HDW = 8Ah (10001010b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	0	1	0	1	0	command
w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	parm1

Description: **HideWindows** causes all specified and currently defined windows to be removed from the receiver display screen. For example, a *window map* value of 72 (hex) hides windows 6 (w_6), 5 (w_5), 4 (w_4), and 1 (w_1). A *window map* value of FF (hex) causes all existing windows in the receiver to be hidden.

Exhibit 15 Page 261

SONY0005778

```
┌─────────────────────────────────────┐
│      TOGGLE WINDOWS - (TGW)          │
└─────────────────────────────────────┘
```

Name: **Toggle Windows** - Toggles Display/Hide status of a set of windows

Command Type: Window

Format: **Toggle Windows** (*window map*)

Parameters: ■ *window map* is an 8-bit bitmap which specifies the window(s) affected by the command. Each bit position represents the window (i.e., *window ID*) to be affected (e.g., bit position 4 addresses the window with the *window ID* of 4). A value of 1 in a bit position specifies that the associated window is to be processed by the command. A value of 0 indicates that the associated window is unaffected by the command.

Command Coding: **TGW = 8Bh (10001011b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	0	1	0	1	1	command
w_7	w_6	w_5	w_4	w_3	w_2	w_1	w_0	parm1

Description: **Toggle Windows** causes all specified and currently defined windows to toggle their display/hide status. That is, the specified windows in the *window map* which are currently displayed will be hidden, and the hidden ones will be displayed. For example, a *window map* value of 83 (hex) toggles windows 7 (w_7), 1 (w_1), and 0 (w_0). A *window map* value of FF (hex) causes all existing windows in the receiver to be toggled.

Exhibit 15 Page 262

SONY0005779

SET WINDOW ATTRIBUTES - (SWA)

Name: SetWindowAttributes - Defines the window styles for the current window.

Command Type: Window

Format: **SetWindowAttributes** (*justify, print direction, scroll direction, wordwrap, display effect, effect direction, effect speed, fill color, fill opacity, border type, border color*)

Parameters:
■ *justify* (j) specifies how the text to be written in the window will be justified. [LEFT, RIGHT, CENTER, FULL] == [0, 1, 2, 3].

■ *print direction* (pd) specifies in which direction text will be written in the window. [LEFT_TO_RIGHT, RIGHT_TO_LEFT, TOP_TO_BOTTOM, BOTTOM_TO_TOP] == [0, 1, 2, 3].

■ *scroll direction* (sd) specifies which direction text will scroll when the end of a caption "line" is reached. [LEFT_TO_RIGHT, RIGHT_TO_LEFT, TOP_TO_BOTTOM, BOTTOM_TO_TOP] == [0, 1, 2, 3]..

■ *wordwrap* (ww), when set to YES, word wrapping is enabled. When set to NO, wordwrapping is disabled. [YES, NO] == [1, 0].

■ *display effect* (de) specifies the effect that is to take place when the window is displayed and when it is hidden. When the SNAP effect is chosen, the window will pop-on the screen when the window is displayed and pop-off when the window is hidden. The FADE effect causes the window to fade onto and off of the screen at the specified *effect rate*. The WIPE effect causes the window to swipe onto and off of the screen at the specified *effect rate* and *effect direction*. [SNAP, FADE, WIPE] == [0, 1, 2].

■ *effect direction* (ed) specifies which direction a WIPE window will appear on the screen. Note that a WIPE window will wipe-off of the screen in the opposite direction from which it wiped-on. [LEFT_TO_RIGHT, RIGHT_TO_LEFT, TOP_TO_BOTTOM, BOTTOM_TO_TOP] == [0, 1, 2, 3].

■ *effect speed* (es) specifies, in .5 second units, how fast windows with WIPE and FADE effects will appear and disappear from the screen when they are displayed and hidden. The effect speed value may range from 1 to 15, approximating .5 (1 x .5) to 7.5 (15 x .5) seconds of effect speed variation.

■ *fill color* (fr, fg, fb) is the color of the windows interior (see Section 8.4.11).

■ *fill opacity* (fo) is the characteristic of the fill color of the window and the window border. [SOLID, FLASH, TRANSLUCENT, TRANSPARENT] == [0, 1, 2, 3].

■ *border type* (bt) defines the type of outer edge surrounding the window. [NONE, RAISED, DEPRESSED, UNIFORM, SHADOW_LEFT, SHADOW_RIGHT] == [0, 1, 2, 3, 4, 5].

■ *border color* (br, bg, bb) is the color of the windows outer edge(see Section 8.4.11).

48

Exhibit 15 Page 263

SONY0005780

Command Coding: **SWA = 97h (10010111b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	1	0	1	1	1	command
fo_1	fo_0	fr_1	fr_0	fg_1	fg_0	fb_1	fb_0	parm1
bt_1	bt_0	br_1	br_0	bg_1	bg_0	bb_1	bb_0	parm2
bt_2	ww	pd_1	pd_0	sd_1	sd_0	j_1	j_0	parm3
es_3	es_2	es_1	es_0	ed_1	ed_0	de_1	de_0	parm4

Description:

SetWindowAttributes assigns the specified style attributes to the current window. This command can be issued any number of times to an existing window. The style attributes will overwrite any existing attributes assigned to the window.

49

Exhibit 15 Page 264

SONY0005781

SET PEN ATTRIBUTES - (SPA)

Name: **SetPenAttributes** - Assign pen style attributes for the current window.

Command Type: Pen

Format: **SetPenAttributes** (*pen size, font, text tag, offset, italics, underline, edge type*)

Parameters:

■ *pen size* (s) defines which of three pen sizes is to be used for text written to the current window, as specified by the current window ID. Note that the pen size displayed on the screen can be overridden by the user. [SMALL, STANDARD, LARGE] == [0, 1, 2].

■ *font style* (fs) specifies which one of 8 different predefined font styles (see Section 8.5.3) to be used for text written to the current window (0 - 7).

     0 - Default (undefined)
     1 - Monospaced with serifs
     2 - Proportionally spaced with serifs
     3 - Monospaced without serifs
     4 - Proportionally spaced without serifs
     5 - Casual font type
     6 - Cursive font type
     7 - Small capitals

■ *text tag* (tt) specifies which one of 16 different predefined caption text function tags (see Section 8.5.9) is to be associated with the following caption text to be written to the current window (0 - 15).

     0 - Dialog
     1 - Source or speaker ID
     2 - Electronically reproduced voice
     3 - Dialog in language other than primary
     4 - Voiceover
     5 - Audible Translation
     6 - Subtitle Translation
     7 - Voice quality description
     8 - Song Lyrics
     9 - Sound effect description
     10 - Musical score description
     11 - Expletive
     12 to 14 - (undefined)
     15 - Text not to be displayed

■ *offset* (o) specifies sub-scripting and super-scripting attributes for text written to the current window. [SUBSCRIPT, NORMAL, SUPERSCRIPT] == [0, 1, 2].

■ *italics* (i) specifies if text written to the current window is italicized. [YES, NO] == [1, 0].

■ *underline* (u) specifies if text written to the current window is underlined. [YES, NO] == [1, 0].

■ *edge type* (et) is the type of outlined edge of the text. [NONE, RAISED, DEPRESSED, UNIFORM, LEFT_DROP_SHADOW, RIGHT_DROP_SHADOW] == [0, 1, 2, 3, 4, 5].

Exhibit 15 Page 265

SONY0005782

Command Coding:　　**SPA = 90h (10010000b)**

b₇	b₆	b₅	b₄	b₃	b₂	b₁	b₀	
1	0	0	1	0	0	0	0	command
tt_3	tt_2	tt_1	tt_0	o_1	o_0	s_1	s_0	parm1
i	u	et_2	et_1	et_0	fs_2	ft_1	ft_0	

Description:

SetPenAttributes assigns pen style attributes for the currently defined window. Text written to the current window will have the attributes specified by the most recent **SetPenAttributes** command written to the window. Pen attributes for a window can be changed as often as desired. These attributes will remain in effect for the window during its entire existence.

51

Exhibit 15 Page 266

SONY0005783

SET PEN COLOR - (SPC)

Name: **SetPenColor** - Assign styles to a dynamic preset style number.

Command Type: Pen

Format: **Set Pen Color** (fg *color, fg opacity, bg color, bg opacity, edge color)*

Parameters:

■ *fg color* (fr, fg, fb) is the color of the text foreground body (see Section 8.5.6).

■ *fg opacity* (fo) is the characteristic of the text foreground body. [SOLID, FLASH, TRANSLUCENT, TRANSPARENT] ═══ [0, 1, 2, 3].

■ *bg color* (br, bg, bb) is the color of the background box surrounding the window text (see Section 8.5.7).

■ *bg opacity* (bo) is the characteristic of the text background. [SOLID, FLASH, TRANSLUCENT, TRANSPARENT] ═══ [0, 1, 2, 3].

■ *edge color* (er, eg, eb) is the color of the outlined edges of the text. The text character edges have the same opacity value as *fg opacity* (see Section 8.5.6).

Command Coding: **SPC = 91h (10010001b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	1	0	0	0	1	command
fo_1	fo_0	fr_1	fr_0	fg_1	fg_0	fb_1	fb_0	parm1
bo_1	bo_0	br_1	br_0	bg_1	bg_0	bb_1	bb_0	parm2
0	0	er_1	er_0	eg_1	eg_0	eb_1	eb_0	parm3

Description:

SetPenColor assigns the pen color attributes for the current window, as specified by the current window ID. Text written to the current window will have the color attributes specified by the most recent **SetPenColor** command written to the window. Pen color attributes for a window can be changed as often as desired. These attributes will remain in effect for the window during its entire existence.

52

Exhibit 15 Page 267

SONY0005784

SET PEN LOCATION - (SPL)

Name: SetPenLocation - Specifies the pen cursor location within a window.

Command Type: Pen

Format: SetPenLocation *(row, column)*

Parameters:
- *row* (r) is the text row within the current window's text buffer   (0-14).
- *column* (c) is the text column within the current window's text buffer (0-31 for 4:3 formats, 0-41 for 16:9 formats).

Command Coding: SPL = 92h (10010010b)

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	1	0	0	1	0	command
0	0	0	0	r_3	r_2	r_1	r_0	parm1
0	0	c_5	c_4	c_3	c_2	c_1	c_0	parm2

Description: SetPenLocation repositions the pen cursor for the current window, as specified by the current window ID. . When the window justification type is "left," The next group of text written to the current window will start at the specified row and column, and justification will be ignored. When the window justification type is not left and the print direction is left-to-right or right-to-left, the column parameter shall be ignored. When the window justification type is not left and the print direction is top-to-bottom or bottom-to-top, the row parameter shall be ignored. When the window justification type is not left, text shall be formatted based upon the current window justification type. Note that if a window is not *locked* (see **DefineWindow**) and the SMALL *pen size* (see **SetPenAttributes**) is in effect, more than 12 rows and 36 columns could possibly be addressed.

53

Exhibit 15 Page 268

SONY0005785

DELAY - (DLY)

Name: Delay - Delays service data interpretation

Command Type: Synchronization

Format: **Delay** *(tenths of seconds)*

Parameters: ■ *tenths of seconds* (t) is the number of tenths of seconds to delay before recommencing service data interpretation.

Command Coding: **DLY = 8Dh (10001101b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	0	1	1	0	1	command
t_7	t_6	t_5	t_4	t_3	t_2	t_1	t_0	parm1

Description: **Delay** instructs receivers to suspend interpretation of the current service's command input buffer. The delay is specified in tenths of seconds. Once the delay time expires, interpretation of caption commands recommences.

The delay value may range from 1 to 255 -- which specifies an effective delay time from 1/10 to 25.5 (255/10) seconds.

A delay for a service will remaining in effect until one of the following occurrences:

- the specified delay time expires
- a DelayCancel command is received
- the service's input buffer becomes full
- a service Reset command is received

54

Exhibit 15 Page 269

SONY0005786

DELAY CANCEL - (DLC)

Name: **DelayCancel** - Cancels an Active Delay Command

Command Type: Synchronization

Format: **DelayCancel**

Parameters: none

Command Coding: **DLC = 8Eh (10001110b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	0	1	1	1	0	command

Description: **DelayCancel** command terminates any active **Delay** command processing within the decoder.

55

Exhibit 15 Page 270

SONY0005787

RESET - (RST)

Name: **Reset** - Resets the Caption Channel Service

Command Type: Synchronization

Format: **Reset**

Parameters: none

Command Coding: **RST = 8Fh (10001111b)**

b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0	
1	0	0	0	1	1	1	1	command

Description: **Reset** command reinitializes the service for which it is received.

56

Exhibit 15 Page 271

SONY0005788

9 DTVCC Decoder Manufacturer Recommendations
The following are the recommendations for DTV Closed Captioning decoder implementation including DTV, SDTVand HDTV. These recommendations are directed to the least common denominator of all of the DTVCC features described in the previous sections. Although voluntary, these recommendations should be considered as requirements for a realistic minimal implementation of the DTVCC capabilities. These minimal recommendations provide a bridge from NTSC (EIA-608-A) captioning implementation to the eventual full-feature implementation of this DTVCC specification .

It should be emphasized that these minimum recommendations are not intended to, and should not, restrict caption providers from using the whole suite of DTVCC commands and their extensive capabilities. The following sections address the minimum recommendations that have been anticipated, but may not cover all conditions and manifestations. It is up to the manufacturer to consider all situations that are not explicitly presented herein.

> NOTE--The section numbers in the following headings refer to the corresponding sections in the current DTVCC Specification, EIA-708-B, to which the minimum recommendations apply.

9.1 DTVCC Section 4.2 - Pre-Allocated Bandwidth
While the DTVCC Caption Channel provides a continuous 9600 bps bit stream within the DTVCC Transport Channel, the individual bandwidth allocated to any single service shall not exceed 25% of the total bandwidth averaged over any 1 second time interval. This limit permits a maximum, average captioning data rate of 300 Bps per service.

That is, decoders need only implement enough buffering and processing power to handle a maximum of 2400 bps for each service. In effect, when this limit is exceeded for a service, the input storage buffer allocated for the service will overflow and data not already buffered will be lost.

> NOTE-- In contrast, the per-service limitation addressed above still provides a five-fold enhancement over the maximum possible NTSC Closed-Caption service data rate of 60 Bps.

9.2 DTVCC Section 6.1 - Services
Decoders should be capable of decoding and processing data for at least one (1) service. Decoders shall be capable of decoding and processing the Caption Service Directory data.

9.3 DTVCC Section 6.2 - Caption Channel Service Blocks
Decoders should be capable of decoding all Caption Channel Block Headers consisting of Standard Service Headers, Extended Service Block Headers, and Null Block headers. However, decoding of the data is required only for Standard Service Blocks (Service IDs <= 6), and then only if the characters for the corresponding language are supported.

Decoders should be able to display the directory for services 1 through 6. Service decoding and directory display for services numbered 7 or greater are optional.

9.4 DTVCC Section 7.1 - Code Space Organization
Decoders must support Code Space C0, G0, C1, and G1 in their entirety.

The following characters within code space G2 must be supported:

- transparent space ($_{TSP}$)
- non-breaking transparent space ($_{NBTSP}$)
- solid block (■)
- trademark symbol (TM)
- Latin-1 characters (Š, Œ, š, œ, Ÿ)

The substitutions in Table 17 are to be made if a decoder does not support the remaining G2 characters.

57

Exhibit 15 Page 272

SONY0005789

G2 Character	Substitute With
open single quote ('), G2 char code 0x31	G0 single quote ('), char code 0x27
close single quote('), G2 char code 0x32	G0 single quote ('), char code 0x27
open double quote ("), G2 char code 0x33	G0 double quote ("), char code 0x22
close double quote ("), G2 char code 0x34	G0 double quote ("), char code 0x22
bold bullet (•), G2 char code 0x35	G1 bullet (·), char code 0xB7
ellipsis(...), G2 char code 0x25	G0 underscore (_), char code 0x5F
one-eighth (¹/₈), G2 char code 0x76	G0 percent sign (%), char code 0x25
three-eighths (³/₈), G2 char code 0x77	G0 percent sign (%), char code 0x25
five-eighths (⁵/₈), G2 char code 0x78	G0 percent sign (%), char code 0x25
seven-eighths (⁷/₈), G2 char code 0x79	G0 percent sign (%), char code 0x25
vertical border ( ), G2 char code 0x7A	G0 stroke (|), char code 0x7C
upper-right border ( ), G2 char code 0x7B	G0 dash (-), char code 0x2D
lower-left border (L), G2 char code 0x7C	G0 dash (-), char code 0x2D
horizontal border (—), G2 char code 0x7D	G0 dash (-), char code 0x2D
lower-right border ( ), G2 char code 0x7E	G0 dash (-), char code 0x2D
upper-left border ( ), G2 char code 0x7F	G0 dash (-), char code 0x2D

Table 17 G2 Character Substitution Table

Support for code spaces C2, C3, and G3 is optional.

All unsupported graphic symbols in the G3 code space are to be substituted with the G0 underscore character (_), char code 0x5F.

9.5 DTVCC Section 8.2 - Screen Coordinates

Table 18 specifies the screen coordinate resolutions and limits for anchor point positioning in 4:3 and 16:9 display formats, and the number of characters per row.

Screen Aspect Ratio	Maximum Anchor Position Resolution	Minimum Anchor Position Resolution	Maximum Displayed Rows	Maximum Characters per Row
4:3	75v x 160h	15v x 32h	4	32
16:9	75v x 210h	15v x 42h	4	42
other	75v x (5 x H)	15v x H*	4	*

Table 18 Screen Coordinate Resolutions & Limits

*H = 32 x (the width of the screen in relation to a 4:3 display). For example, the 16:9 format is 1/3 wider than a 4:3 display; thus, H = 32 * 4/3 = 42.667, or 42.

This means that the minimum grid resolution for a 4:3 aspect ratio instrument is 15 vertical positions x 32 horizontal positions. This minimum grid resolution for 16:9 ratio instrument is 15 vertical positions x 42 horizontal positions. These minimum grid sizes are to cover the entire safe-title area of the corresponding screen.

The minimum coordinates equate to a 1/5 reduction in the maximum horizontal and vertical grid resolution coordinates. Caption providers are to use the maximum coordinate system values when specifying anchor point positions. Decoders using the minimum resolution are to divide the provided horizontal and vertical screen coordinates by 5 to derive the equivalent minimum coordinates.

Any caption targeted for both 4:3 and 16:9 instruments is limited to 32 contiguous characters per row. If a caption is received by a 4:3 instrument that is targeted for a 16:9 display only, or requires a window width greater than 32

58

Exhibit 15 Page 273

SONY0005790

characters, then the caption may be completely disregarded by the decoder. 16:9 instruments should be able to process and display captions intended for 4:3 displays, providing all other minimum recommendations are met.

If the resulting size of any window is larger than the safe title area for the corresponding display's aspect ratio, then this window will be completely disregarded.

9.6 DTVCC Section 8.4 - Caption Windows

Decoders need to display no more than 4 rows of captions on the screen at any given time, regardless of the number of windows displayed. This implies that no more than 4 windows can be displayed at any given time (with each having only one caption row).

However, decoders should maintain storage to support a minimum total of 8 rows of captions. This storage is needed for the worst-case support of a displayed window with 4 rows of captioning and a non-displayed window which is buffering the incoming rows for the next 4-row caption.

As implied above, the maximum number of windows that may be displayed at any one time by a minimum decoder implementation is 4. If more than 4 windows are defined in the caption stream, the decoder may disregard the youngest and lowest priority window definition(s). Caption providers must be aware of this limitation, and either restrict the total number of windows used or accept that some windows will not be displayed.

9.7 DTVCC Section 8.4.2 - Window Priority

Decoders do not need to support overlapped windows. If a window overlaps another window, the overlapped window need not be displayed by the decoder. Decoders may support overlapped windows as an option.

9.8 DTVCC Section 8.4.6 - Window Size

At a minimum, decoders will assume that all windows have rows and columns "locked". This implies that if a decoder implements the optional SMALL pen-size, then word-"un"wrapping, when shrinking captions, need not be implemented. Also, if a decoder implements the optional LARGE pen size, then word wrapping (when enlarging captions) need not be implemented.

9.9 DTVCC Section 8.4.8 - Word Wrapping

Decoders may support word wrapping as an option.

9.10 DTVCC Section 8.4.9 - Window Text Painting

9.10.1 Justification

All decoders should implement "left", "right", and "center" caption-text justification. Implementation of "full" justification is optional. . If "full" justification is not implemented, fully justified captions should be treated as though they are "left" justified.

For "left" justification, decoders should display any portion of a received row of text when it is received. For "center", "right", and "full" justification, decoders may display any portion of a received row of text when it is received, or may delay display of a received row of text until reception of a row completion indicator. A row completion indicator is defined as receipt of a CR, ETX or any other command , except SetPenColor, SetPenAttributes, or SetPenLocation where the pen relocation is within the same row.

Receipt of a character for a displayed row which already contains text with "center", "right" or "full" justification will cause the row to be cleared prior to the display of the newly received character and any subsequent characters. Receipt of a justification command which changes the last received justification for a given window will cause the window to be cleared.

9.10.2 Print Direction

At a minimum, decoders must support LEFT_TO_RIGHT printing.

9.10.3 Scroll Direction

At a minimum, decoders must support BOTTOM_TO_TOP scrolling.

For windows sharing the same horizontal scan lines on the display, scrolling may be disabled.

Exhibit 15 Page 274

SONY0005791

9.10.4 Scroll Rate
At a minimum, decoders must support the same recommended practices for scroll rate as is provided for NTSC closed-captioning.

9.10.5 Smooth Scrolling
At a minimum, decoders must support the same recommended practices for smooth scrolling as is provided for NTSC closed-captioning.

9.10.6 Display Effects
At a minimum, decoders must implement the "snap" window display effect. If the window "fade" and "wipe" effects are not implemented, then the decoder will "snap" all windows when they are to be displayed, and the "effect speed" parameter is ignored.

9.11 DTVCC Section 8.4.11 - Window Colors and Borders
At a minimum, decoders need only to implement borderless windows with solid, black backgrounds (i.e., border type = NONE, fill color = (0,0,0), fill opacity = SOLID), and borderless transparent windows (i.e., border type = NONE, fill opacity = TRANSPARENT).

9.12 DTVCC Section 8.4.12 - Predefined Window and Pen Styles
Predefined Window Style and Pen Style ID's may be provided in the DefineWindow command. At a minimum, decoders should implement Predefined Window Attribute Style 1 and Predefined Pen Attribute Style 1, as shown in Table 19 and Table 20.

Exhibit 15 Page 275

SONY0005792

Style ID #	Justify	Print Direction	Scroll Direction	Word Wrap	Display Effect	Effect Direction	Effect Speed	Fill Color	Fill Opacity	Border Type	Border Color	Usage
1	LEFT	LEFT-TO-RIGHT	BOTTOM-TO-TOP	NO	SNAP	n/a	n/a	(0,0,0) Black	SOLID	NONE	n/a	NTSC Style PopUp Captions
2	LEFT	LEFT-TO-RIGHT	BOTTOM-TO-TOP	NO	SNAP	n/a	n/a	n/a	TRANS-PARENT	NONE	n/a	PopUp Captions w/o Black Background
3	CNTR	LEFT-TO-RIGHT	BOTTOM-TO-TOP	NO	SNAP	n/a	n/a	(0,0,0) Black	SOLID	NONE	n/a	NTSC Style Centered PopUp Captions
4	LEFT	LEFT-TO-RIGHT	BOTTOM-TO-TOP	YES	SNAP	n/a	n/a	(0,0,0) Black	SOLID	NONE	n/a	NTSC Style RollUp Captions
5	LEFT	LEFT-TO-RIGHT	BOTTOM-TO-TOP	YES	SNAP	n/a	n/a	n/a	TRANS-PARENT	NONE	n/a	RollUp Captions w/o Black Background
6	CNTR	LEFT-TO-RIGHT	BOTTOM-TO-TOP	YES	SNAP	n/a	n/a	(0,0,0) Black	SOLID	NONE	n/a	NTSC Style Centered RollUp Captions
7	LEFT	TOP-TO-BOTTOM	RIGHT-TO-LEFT	NO	SNAP	n/a	n/a	(0,0,0) Black	SOLID	NONE	n/a	Ticker Tape

Table 19 Predefined Window Style ID's

61

EIA-708-B

Predefined Style ID	Pen Size	Font Style	Offset	Italics	Underline	Edge Type	Foregrnd Color	Foregrnd Opacity	Backgrnd Color	Backgrnd Opacity	Edge Color	Usage
1	STNDR	0	NORMAL	NO	NO	NONE	(2,2,2) White	SOLID	(0,0,0) Black	SOLID	n/a	*Default NTSC Style**
2	STNDR	1	NORMAL	NO	NO	NONE	(2,2,2) White	SOLID	(0,0,0) Black	SOLID	n/a	*NTSC Style* Mono w/ Serif*
3	STNDR	2	NORMAL	NO	NO	NONE	(2,2,2) White	SOLID	(0,0,0) Black	SOLID	n/a	*NTSC Style* Prop w/ Serif*
4	STNDR	3	NORMAL	NO	NO	NONE	(2,2,2) White	SOLID	(0,0,0) Black	SOLID	n/a	*NTSC Style* Mono w/o Serif*
5	STNDR	4	NORMAL	NO	NO	NONE	(2,2,2) White	SOLID	(0,0,0) Black	SOLID	n/a	*NTSC Style* Prop w/o Serif*
6	STNDR	3	NORMAL	NO	NO	UNIFRM	(2,2,2) White	SOLID	n/a	TRANS-PARENT	(0,0,0) Black	*Mono w/o Serif, Bordered Text, No BG*
7	STNDR	4	NORMAL	NO	NO	UNIFRM	(2,2,2) White	SOLID	n/a	TRANS-PARENT	(0,0,0) Black	*Prop. w/o Serif, Bordered Text, No BG*

Table 20 Predefined Pen Style ID's

* "NTSC Style" - White Text on Black Background

62

9.13 DTVCC Section 8.5.1 - Pen Size
At a minimum, decoders must support the STANDARD pen size, with the implementation of the LARGE and SMALL pen sizes being optional.

The STANDARD pen size should be implemented such that the height of the tallest character in any implemented font is no taller than 1/15 of the height of the safe-title area, and the width of the widest character is no wider than 1/32 of the width of the safe-title area for 4:3 displays and 1/42 of the safe-title area width for 16:9 displays.

The LARGE pen size should be implemented such that the width of the widest character in any implemented font is no wider than 1/32 of the safe-title area for 16:9 displays. This recommendation allows for captions to grow to a LARGE pen size without having to reformat the caption since no caption will have more than 32 characters per row (see Section 8.4.6).

9.14 DTVCC Section 8.5.3 - Font Styles
Although a caption service provider may specify any one of 8 font styles using the **SetPenAttributes** command, decoders need only to implement a single font for caption text display.

Decoders that implement more than one font but do not support a font style specified in the **SetPenAttributes** command should instead display the caption text in the most similar font available. In decoders with only one font (i.e., font style 0, the default), all caption text, regardless of the specified font style, will be displayed in the default font.

In decoders with more than one but less than eight fonts, unsupported font styles should be displayed using an alternate font, giving precedence to the spacing attribute of the indicated font style, if possible. For example, if the specified but unsupported font style is "monospaced with serifs", the best substitute would be another monospaced font, and the second-best alternative would be a proportionally spaced font with serifs. If the Cursive font style is not supported, an acceptable substitution is an italicized version of an available font.

All supported font styles may be implemented in any typeface which the decoder manufacturer deems to be a readable rendition of the font style, and need not be in the exact typefaces given as examples in Section 8.5.3.

9.15 DTVCC Section 8.5.4 - Character Offsetting
Decoders need not to implement the character offsetting (i.e., subscript and superscript) pen attributes.

9.16 DTVCC Section 8.5.5 - Pen Styles
At a minimum, decoders must implement normal, italic, and underline pen styles.

9.17 DTVCC Section 8.5.6 - Foreground Color and Opacity
At a minimum, decoders must implement solid and flashing character foreground type attributes.

At a minimum, decoders must implement the following character foreground colors: white, black, red, green, blue, yellow, magenta and cyan.

9.18 DTVCC Section 8.5.7 - Background Color and Opacity
Decoders need only implement solid black character backgrounds. It is recommended that this background is extended beyond the character foreground to a degree that the foreground is separated from the underlying video by a sufficient number of background pixels to insure the foreground is separated from the background.

9.19 DTVCC Section 8.5.8 - Character Edges
Decoders need not to implement separate character edge color, opacity, and type attribute control. In this case, there is no separately controlled edge surrounding the body of characters.

9.20 DTVCC Section 8.8 - Color Representation
At a minimum, decoders must support the 8 colors described in Table 21.

63

Exhibit 15 Page 278

SONY0005795